THESE DE DOCTORAT

présentée

A L'UNIVERSITE PARIS 7

Spécialité : Informatique Fondamentale

par

Hugo HERBELIN

Sujet de la thèse :

Séquents qu'on calcule

De l'interprétation du calcul des séquents comme calcul de λ-termes et comme calcul de stratégies gagnantes

Soutenue le 23 Janvier 1995 devant la Commission d'examen composée de

MM.	**Gérard HUET**	**Président**
	Stefano BERARDI	**Rapporteurs**
	René DAVID	
	Michel PARIGOT	
	Thierry COQUAND	**Directeur**
	Serge GRIGORIEFF	**Examinateur**

Remerciements

Pourrait-on qualifier cette thèse de “téléthèse” ? Son superviseur, en l'occurrence Thierry Coquand (ci-après dénommé Thierry pour plus de simplicité), ayant en effet prévenu, dès le départ, que Göteborg, en Suède, serait son port d'attache des prochaines années.

Thierry faisait partie de l'équipe Formel de l'INRIA et c'est là-bas que j'ai commencé ma thèse, dans l'environnement de Gérard Huet, homme dont la rigueur et les qualités d'encadrement continuent de m'impressionner et qui m'a, tout au long de ma thèse, toujours soutenu dans mes efforts et mes démarches, qui m'a aidé aussi dans le financement de mes pèlerinages revigorants en Suède et à qui revient tout naturellement la position de président du jury de ma soutenance.

Très vite, c'est en échange d'enseignements à Lyon que j'ai pu être subventionné pour ma thèse. Ce fut un moment d'écartèlement maximum : Thierry à l'étranger, mon enseignement à Lyon, moi “paumé” de banlieue sud-parisienne. Si j'ai survécu tant bien que mal, c'est grâce à la patience et à la confiance de Christine Paulin-Mohring et grâce aux relations que j'ai pu garder avec ce que j'appelais l'équipe Formel, c'est-à-dire, outre Gérard Huet, avec Gilles Dowek et Benjamin Werner, thésards comme moi à l'époque, avec Daniel de Rauglaudre, “the french meharist”, puis plus tard, avec Samuel Boutin, Amokrane Saibi et Chet Murthy, sans oublier Paul-André Méllies, bien que membre d'une équipe “parallèle”.

Je reçus avec joie l'annonce de mon “Attachement Temporaire” à l'UFR d'informatique de Paris 7. De la part de Serge Grigorieff, j'ai eu un soutien actif et attentif et je suis heureux qu'il ait bien voulu, en toute simplicité, participer au jury de ma thèse. Il m'a permis de bénéficier d'un bureau et d'une machine à distance raisonnable de mon lieu de résidence et, grâce à son groupe de travail, j'ai pu commencer à expérimenter mon statut d'associé au LITP, partageant notamment, sur le plan scientifique, avec Loïc Colson, Tristan Crolard, Maurice Margenstern et Pierre Valarcher. Mon environnement était alors de qualité : assis devant ma console en relation directe avec Thierry, bien entouré, gardant des contacts avec l'équipe Formel devenue équipe Coq et m'enrichissant par d'excellentes relations d'enseignement, j'entamais une phase plus épanouie de ma thèse.

A Paris 7, il y a aussi “le cinquième étage”, et, là, quelques mètres à l'aplomb du bureau des logiciens, versant “informatique”, on rencontre les logiciens, versant “mathématiques”. Plus spécialement, j'ai profité là-haut d'échanges avec Pascal Manoury, Philippe Cusson, Paul Rozière, ce discoureur (hum...) de Vincent Danos, Jean-Baptiste Joinet, et, surtout, avec l'indépendant Michel Parigot, qui me fait un réel honneur en voulant bien être rapporteur de ma thèse.

Ma rencontre effective avec Stefano Berardi doit remonter à bientôt 3 ans. Chacun des thèmes évoqués dans cette thèse, il y a travaillé, et, de sa part à lui aussi, c'est un honneur pour moi qu'il ait accepté, bien que de langue italienne, de lire ma thèse dans le texte et d'en faire un rapport.

Alors qu'il travaille dans un domaine de recherche sans doute plus éloigné du mien que ne le sont ceux de Michel Parigot et Stefano Berardi, René David a bien voulu aussi être un de mes rapporteurs. Je l'en remercie d'autant plus qu'il a eu la gentillesse de consacrer un temps non négligeable à m'orienter dans la rédaction de ma thèse.

Enfin, remerciements spéciaux à tous ceux qui se savent concernés par cette phrase.

Introduction

L'informatique et la logique se rejoignent dans leurs fondements : une preuve est un programme. Plus précisément, si A est une formule alors une preuve de A est un programme dont on sait garantir la terminaison de l'exécution, ainsi que la conformité du résultat à la formule A, interprétée comme une spécification. C'est de l'étude de l'identification entre preuves et programmes, dans le cadre du "calcul des séquents", dont il est question ici.

Les premiers systèmes formalisant, à la fin du siècle dernier, le raisonnement mathématique, étaient de type axiomatique : on se donnait un ensemble d'axiomes exprimant les propriétés des diverses notions logiques, et, à partir de cet ensemble d'axiomes, en appliquant la règle de Modus Ponens (si "A" et si "$A \to B$" alors "B"), on déduisait des théorèmes.

Conjointement à la conception de ces systèmes formels, on s'attachait à montrer leur cohérence, c'est-à-dire, leur aptitude à ne pas permettre de prouver simultanément une chose et son contraire. Cependant, la structure des preuves dans les systèmes axiomatiques était peu adaptée à l'étude de la cohérence.

Un progrès substantiel fut obtenue par Gentzen en 1935, lorque celui-ci donna la définition de deux nouveaux types de systèmes formels, basés non plus sur des axiomes mais sur des règles d'inférences ad hoc internalisant le "sens" de chacune des notions logiques.

De fait, une propriété essentielle de ces nouveaux systèmes formels, couramment repris dans la littérature scientifique actuelle sous les noms respectifs de "déduction naturelle" et "calcul des séquents", est de pouvoir associer à toute preuve une forme canonique de cette preuve, dite "forme normale" et caractérisée par le fait qu'aucun recours à des notions non déjà contenues dans l'énoncé de la formule prouvée n'y apparaît.

L'existence d'une telle forme canonique pour les preuves a cette conséquence : lors d'une recherche systématique de preuves pour une certaine formule, on peut restreindre l'espace de recherche à l'espace des preuves normales. Il est alors aisé de déduire la cohérence de ces systèmes : une simple analyse par cas montre qu'un énoncé et son contraire ne peuvent avoir simultanément une forme normale, et donc, l'un des deux au moins n'est pas prouvable.

La situation était contrastée entre les deux calculs de Gentzen. Une motivation pour la déduction naturelle était de reproduire assez fidèlement le raisonnement intuitif, mais ce n'est que pour le calcul des séquents qu'une notion de preuve normale fut caractérisée par Gentzen. Cette notion de preuve normale est simple (elle revient à interdire une certaine règle, dite de coupure, dont le rôle est à peu près similaire à celui de Modus Ponens) et Gentzen décrit comment, par élimination des coupures, on peut obtenir une preuve normale à partir d'une preuve quelconque. On peut ainsi dire que le calcul des séquents est le premier formalisme logique à avoir été vu comme un "calcul" à proprement parler.

L'étude de la déduction naturelle remonte à Prawitz [50] en 1965. On peut y définir relativement aisément une notion de normalité et une notion de mise en forme normale, toutes deux acceptables, du moins dans le cas de la déduction naturelle intuitionniste (i.e. rejetant l'axiome $A \vee \neg A$), comme "canoniques".

Parallèllement à ces questions de fondement de la logique, des travaux virent le jour pour fonder la calculabilité. Les machines de Turing et les fonctions récursives sont bien connues, mais d'autres modèles existent. Schönfinkel [52], en 1925, et surtout Curry [12] mirent au point la logique combinatoire. Church [8], quant à lui, introduisit un modèle assez proche : le λ-calcul.

Logique combinatoire et λ-calcul sont intimement liés à la prouvabilité. Une première remarque de Curry [13], en 1958, mit en évidence une identité de structure entre les "programmes" de la logique combinatoire et celle des preuves des versions intuitionnistes des systèmes axiomatiques.

Plus tard — l'explicitation fut donnée par Howard [31] et de Bruijn —, on remarqua une identité du même ordre entre la structure des termes du λ-calcul et celle des preuves de la déduction naturelle intuitionniste.

Plus expressif et plus intuitif que la logique combinatoire, le λ-calcul retint plus l'attention. La déduction naturelle put alors être pensée comme un langage de "programmes", et la procédure de mise en forme normale trouva une justification opérationnelle à travers son pendant en λ-calcul. Cette approche déboucha sur la conception de langages mêlant preuves et programmes (Théorie des Types de Martin-Löf, Calcul des Constructions, Arithmétique Fonctionnelle d'ordre 2, ...) et mena même à l'implémentation de systèmes consistants de "programmation par preuves" (NuPrl, Coq, ...).

La correspondance de Curry et Howard-de Bruijn n'était, jusque-là, établie que pour le fragment intuitionniste. C'est à partir de travaux de spécialistes de programmation fonctionnelle qu'apparut une solution pour étendre la correspondance au cas de la logique classique, i.e. à une logique ne rejetant pas l'axiome $A \vee \neg A$. Dans le courant des années 80, Felleisen [16] considéra une extension du λ-calcul avec un opérateur appelé C, permettant la manipulation de continuations. Griffin [29] s'aperçut un peu plus tard que si on transposait cet opérateur C sur un plan logique, il donnait un sens calculatoire à l'axiome $\neg\neg A \to A$ (cet axiome est équivalent à l'axiome $A \vee \neg A$). Murthy [45] puis Krivine éclaircirent le sens de cette correspondance.

Une autre approche a été suivie par Parigot [48] qui mit au point une version classique de la déduction naturelle pour laquelle le côté classique du calcul n'était pas obtenu par l'axiome $A \vee \neg A$ mais en considérant un vecteur de conclusions. Une extension du λ-calcul est associée à cette version classique de la déduction naturelle, c'est le $\lambda\mu$-calcul.

Mais la logique classique, comme il apparaît dans la procédure d'élimination des coupures de Gentzen, a un aspect non-déterministe. Des calculs mettant en valeur ce non-déterminisme ont été conçus, par exemple le λ_{sym}-calcul de Berardi et Barbanera [3].

Conjointement au développement de l'étude de l'aspect calculatoire de la logique classique, motivée par le fait qu'en calcul des séquents l'axiome $A \vee \neg A$ s'obtient "magiquement" (cf Gentzen [20]) par une astuce de syntaxe, une préoccupation nouvelle a émergé pour associer un langage de termes au calcul des séquents et en tirer une justification "opérationnelle" de l'élimination des coupures (cf par exemple Breazu-Tanen [7], Wadler [59], ...). L'un des problèmes, dans une première étape, est d'analyser la part d'arbitraire dans l'élimination des coupures.

C'est dans cette optique que se situe le premier chapitre de cette thèse. Les deuxième à quatrième chapitres parleront d'une extension de la correspondance de type Curry-Howard au cadre du calcul des séquents. Quant aux chapitres cinq et six, ils étudieront une interprétation du calcul des séquents en terme de jeux.

Pour étendre la correspondance de Curry-Howard-de Bruijn au calcul des séquents, il s'avère qu'il est préférable de considérer des restrictions des calculs des séquents de Gentzen, restrictions dont les preuves normales (i.e. sans coupures) ont la propriété d'être en bijection avec les preuves normales de déduction naturelle — et donc aussi avec les λ-termes normaux. Associer un langage de λ-termes à ces restrictions des versions originales des calculs des séquents est simple : il suffit d'inverser la manière de considérer l'application dans le λ-calcul. Plus précisément, pour se conformer à la structure des preuves en calculs des séquents, il suffit de considérer les λ-termes de la forme $(...(u_0\ u_1)\ ...\ u_n)$ sous la forme $(u_0\ [u_1; ...; u_n])$, c'est-à-dire, comme la paire d'une fonction (le terme u_0) et de la liste de ses arguments (la liste $[u_1; ...; u_n]$).

Par ailleurs, nous accorderons une place explicite à la substitution dans la syntaxe : lorsque les termes du λ-calcul sont définis comme des arbres, l'opération de substitution d'un argument à une variable dans le corps de la fonction nécessite un travail de propagation à travers les termes. Nous ne considèrerons pas comme implicite ce travail et, au contraire, lorsque nous parlerons de λ-calcul, nous associerons explicitement une construction syntaxique à cette opération, dans l'esprit des $\lambda\sigma$-calculs de Curien [11], Lévy-Hardin [30],... ou plutôt de celui de Lescanne [2], car, comme dans ce dernier calcul, l'opération de substitution que nous considèrerons sera un simple "let ... in ..." à la ML.

Ce choix de considérer explicitement un opérateur de substitution apparaîtra d'autant plus justifié dans notre cadre qui est celui du calcul des séquents que l'interprétation naïve de la règle de coupure en termes fonctionnels est précisément d'être un opérateur de substitution.

La correspondance que nous proposons sera valable autant sur un plan intuitionniste que sur un plan classique (en considérant alors le $\lambda\mu$-calcul de Parigot) ; elle s'étendra directement aux connecteurs $\wedge$ et $\vee$.

L'extension à des système de types plus sophistiqués dont la base reste le λ-calcul ne posera pas non plus, a priori, de problèmes.

Elle tend à montrer que le calcul des séquents, quitte à accepter de contrôler une part de l'arbitraire que la version originelle de Gentzen offre, n'est pas moins sujet à une interprétation fonctionnelle que ne l'est la déduction naturelle. Et l'un, et l'autre, sont susceptibles d'être vus comme des calculs de λ-termes, avec, à la clé, une définition canonique de forme normale et l'existence d'une procédure claire de mise en forme normale. Si la déduction naturelle a l'avantage d'être "naturelle", le calcul des séquents a celui d'être plus précis (dans le sens où les règles d'élimination de la déduction naturelle peuvent se décomposer, dans le cadre du calcul des séquents, en une règle d'introduction gauche suivie d'une règle de coupure, et dans le sens aussi, où une application $(...(x\ u_1)\ ...\ u_n)$ n'est plus une forme normale en calcul des séquents, la forme normale étant $(x\ [u_1; ...; u_n]))$. De plus il se prête mieux, comme il l'a toujours fait, à la recherche systématique de preuves.

Nous ne pensons pas que la mise à jour de ce lien entre la structure du calcul des séquents et celle du λ-calcul exclut d'autres interprétations de diverses variantes de calcul des séquents. En particulier, nous étudierons dans une deuxième partie de la thèse une approche du calcul des séquents comme calcul de stratégies gagnantes pour certains jeux à deux joueurs.

Une telle approche est implicite dès la première preuve de cohérence de l'arithmétique de Gentzen (preuve retirée au dernier moment avant impression en raison de critiques sur les moyens de preuves employés, mais maintenant reproduite, accompagnant les autres articles de Gentzen, dans le livre de Szabo [55]). A partir de 1960, des tentatives furent effectuées par Lorenzen [38, 39] et son école dans le but de fonder la prouvabilité intuitionniste sur une notion de jeux à deux joueurs. Cependant, c'est avec Blass [5] en 1991 que l'on peut obtenir une première correspondance règle par règle entre un certain type de jeu (défini par Blass [4]) et la logique linéaire. Peu après, Coquand [9], s'inspirant des travaux de Gentzen, propose une interprétation directe en termes de jeux d'un calcul des séquents classique (dit "le calcul de Novikoff"), et, simultanément, propose une alternative à l'élimination des coupures sous la forme d'un débat finitaire entre deux preuves vues comme des stratégies pour deux joueurs se disputant la validité d'une formule.

Nous montrons deux choses. Premièrement que l'on peut exhiber une variante propositionnelle de LJ (appelée LJQ*) dont l'interprétation canonique en termes de jeux correspond précisément à une sorte de jeux définie par Lorenzen : les E-dialogues. Ceci raffine un résultat d'équivalence de Felscher [17] entre la prouvabilité dans LJ et la possibilité de gagner un E-dialogue. Deuxièmement, que le procédé, développé par Coquand, de laisser deux preuves jouer l'une contre l'autre, est en correspondance opérationnelle avec le procédé d'élimination des coupures du calcul de Novikoff (plus précisément, la correspondance a lieu avec l'élimination des coupures exécutée suivant une stratégie de réduction faible de tête — selon une terminologie empruntée aux implémenteurs de λ-calculs paresseux).

Cette correspondance entre preuves et stratégies gagnantes (dans LJQ*), entre élimination faible de tête des coupures et débat entre preuves (dans le calcul de Novikoff), nous ne la mettons en valeur que dans des calculs bien précis. Nous sommes cependant convaincus que cette correspondance s'étend à d'autres variantes de calculs des séquents, en particulier à ceux décrits dans les chapitres 2 à 4 de la thèse. A la clé, il y a la possibilité d'obtenir une interprétation en terme de jeux du λ-calcul.

Chapitre 1

Calcul des séquents et élimination des coupures

C'est en 1935 que Gentzen [20] donna la définition d'un nouveau type de système formel qualifié par lui de "calcul déductif" (en allemand "Schlußweisenkalkül") de type "calcul logistique" ("Logistischer Kalkül") et aujourd'hui repris sous le nom de "calcul des séquents". Deux versions étaient présentées, l'une, LJ, pour la logique intuitionniste et l'autre, LK, pour la logique classique.

Dans le même article de 1935, Gentzen donnait la définition aussi d'un calcul de la déduction naturelle ("Kalkül des natürlicher Schließens").

La structure arborescente des preuves de la déduction naturelle satisfait la propriété que les feuilles de l'arbre correspondaient à des hypothèses. La structure des preuves des calculs LJ et LK ne partageant pas cette propriété, Gentzen, par opposition, qualifia les calculs LJ et LK de "logistique", au même titre qu'il qualifiait aussi de logistique, et pour les mêmes raisons, les calculs axiomatiques de type Hilbert.

Il y a une autre distinction qui sépare, cette fois, les calculs LJ et LK des deux autres types de systèmes : pour définir LJ et LK, contrairement à ce qui se passe dans les deux autres types de calculs, on ne peut pas se contenter d'une structure arborescente annotée par des formules pour en représenter les preuves. Il faut en effet généraliser la notion de formule à celle de séquent, i.e. de "formule dans un contexte".

Gentzen n'a pas lui-même attribué de qualificatif clair aux calculs LJ et LK. Ces calculs, pour être définis nécessitait la notion de séquent. C'est sous ce nom — le nom de calcul des séquents ("Sequenzkalkül" en allemand, "sequent calculus" en anglais) — qu'au fil des années, ils se sont inscrits dans l'histoire.

Mais autant la distinction entre logistique et non logistique, quelque discutable soit l'intérêt de cette distinction, sépare effectivement LJ et LK de la déduction naturelle, autant l'appellation actuelle de calcul des séquents est tout à fait injustifiée car tout calcul peut s'exprimer comme un calcul de séquents (ce que Gentzen a d'ailleurs fait avec la déduction naturelle pour sa première preuve de cohérence de l'arithmétique [21]).

A défaut de pouvoir prétendre à une meilleure terminologie, nous garderons dans cette thèse la dénomination de "calcul des séquents" (Curry proposait par exemple d'employer le terme de L-système).

1.1 Les calculs LJ et LK de Gentzen

On se restreint au cas de la logique propositionnelle.

1.1.1 Formules

On se donne un ensemble $\mathcal{V}_F$ potentiellement infini de symboles dont les éléments sont appelés **variables propositionnelles** et sont notés par les lettres $X, Y, Z, \ldots$

L'ensemble des **formules** est défini par la grammaire suivante, où X décrit $\mathcal{V}_F$.

$$A \quad ::= \quad X \mid A \wedge A \mid A \vee A \mid A \rightarrow A \mid \neg A$$

$A \wedge B$ représente la **conjonction** des deux formules A et B tandis que $A \vee B$ représente la **disjonction** de ces deux formules. $A \rightarrow B$ représente l'**implication** de la formule B par la formule A et $\neg A$ représente la **négation** de A. Les symboles $\wedge$, $\vee$, $\rightarrow$ et $\neg$ sont des **connecteurs**.

On utilise les lettres $A, B, C, \ldots$ pour désigner les formules.

1.1.2 Séquents

Un **séquent** s'écrit $\Gamma \vdash \Delta$ où Γ et Δ sont des suites finies de formules. Nous appellerons "thèse" le symbole $\vdash$. Les formules de Γ sont appelées **hypothèses** et celles de Δ sont appelées **conclusions**.

1.1.3 Règles d'inférences

On peut différencier plusieurs types de règles d'inférences.

Règles structurelles

Règles d'affaiblissement

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\ \mathit{Aff}_g \qquad\qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\ \mathit{Aff}_d$$

Règles de contraction

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ \mathit{Cont}_g \qquad\qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\ \mathit{Cont}_d$$

Règles d'échange

$$\frac{\Gamma_1, A, B, \Gamma_2 \vdash \Delta}{\Gamma_1, B, A, \Gamma_2 \vdash \Delta}\ \mathit{Ech}_g \qquad\qquad \frac{\Gamma \vdash \Delta_1, A, B, \Delta_2}{\Gamma \vdash \Delta_1, B, A, \Delta_2}\ \mathit{Ech}_d$$

Règle d'axiome

$$\frac{}{A \vdash A}\ \mathit{Ax}$$

Règles d'introduction des connecteurs

.

Règles d'introduction gauche

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge_g^1 \qquad \frac{\Gamma, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge_g^2 \qquad\qquad \frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta}\ \vee_g$$

$$\frac{\Gamma_1 \vdash \Delta_1, A \quad \Gamma_2, B \vdash \Delta_2}{\Gamma_1, \Gamma_2, A \to B \vdash \Delta_1, \Delta_2}\ \to_g \qquad\qquad \frac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\ \neg_g$$

Règles d'introduction droite

$$\frac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, A \vee B}\ \vee_d^1 \qquad \frac{\Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \vee B}\ \vee_d^2 \qquad\qquad \frac{\Gamma \vdash \Delta, A \quad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B}\ \wedge_d$$

$$\frac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta, A \to B}\ \to_d \qquad\qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\ \neg_d$$

Règle de coupure

$$\frac{\Gamma_1 \vdash \Delta_1, A \quad \Gamma_2, A \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ \mathit{Coupe}$$

A est appelée **formule de coupure**.

Remarque : Gentzen rangeait la règle de coupure parmi les règles structurelles. Etant en fait *la* règle de calcul du système, nous préférons la ranger à part.

1.1.4 Preuves

Une **preuve** est un arbre bien fondé formé à partir des règles d'inférence ci-dessus.

Si p est une preuve, on appelle **inférence de tête** ou **règle de tête** la règle se trouvant à la racine de la preuve, voyant celle-ci comme un arbre. Pour une règle d'inférence, on appelle **prémisses** les preuves "du dessus" et on appelle **séquent conclusion** le séquent "du dessous". On appelle **séquent conclusion** d'une preuve le séquent conclusion de sa règle de tête.

1.1.5 Caractérisation des calculs LJ et LK

Le (fragment propositionnel du) calcul LK est le calcul obtenu en considérant l'ensemble des formules, séquents et preuves définis ci-dessous. Le calcul LK est un calcul **classique** : il est complet pour la logique propositionnelle classique.

Le (fragment propositionnel du) calcul LJ est le calcul obtenu en considérant l'ensemble des formules défini ci-dessus, le sous-ensemble des séquents de la forme $\Gamma \vdash \Pi$ avec Π désignant 0 ou 1 formule, ainsi que l'ensemble des preuves ne faisant intervenir que ce type de séquents. Le calcul LJ est un calcul **intuitionniste**.

Remarque : Dans LJ, du fait de la restriction imposée aux séquents (au plus 1 formule à droite du symbole thèse), les règles $Cont_d$ et Ech_d ne sont plus applicables. Quant à la règle Aff_d, elle n'est applicable qu'après une règle $\neg_g$.

1.1.6 Le théorème d'élimination des coupures

Le théorème d'élimination des coupures, appelé *Hauptsatz* par Gentzen exprime que la coupure est une règle superflue des calculs LJ et LK.

Théorème d'élimination des coupures *(Hauptsatz)* : Toute preuve utilisant la règle de coupure peut être transformée en une preuve de même séquent conclusion et n'utilisant pas la règle de coupure.

Ce théorème incite à considérer les preuves sans coupures comme primitives et la règle de coupure comme un schéma de raisonnement *admissible*. C'est l'approche explicitée par Lorenzen dans [37] et qui pousse à formuler l'énoncé suivant, équivalent au précédent :

Théorème d'admissibilité de la coupure : Si les séquents $\Gamma_1 \vdash \Delta_1, A$ et $\Gamma_2, A \vdash \Delta_2$ sont prouvables sans coupures, alors le séquent $\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2$ est prouvable sans coupure.

1.1.7 Variantes de LJ et LK

1- Les connecteurs multiplicatifs

On obtient une variante de LK en remplaçant les règles $\vee_d^1$ et $\vee_d^2$ par cette unique règle :

$$\frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \vee B}\ \vee_d$$

La différence entre cette règle, incompatible avec la logique intuitionniste, et la paire de règles $\vee_d^1$ et $\vee_d^2$ est spécialement mise en évidence dans le cadre de la logique linéaire (cf Girard [23]).

Parallèlement, on peut remplacer les règles $\wedge_g^1$ et $\wedge_g^2$ par cette unique règle :

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge_g$$

A l'opposé de $\vee_d$ cette règle est compatible avec la logique intuitionniste.

2- Les séquents "monolatères" pour la logique classique

En logique propositionnelle classique, on a équivalence entre $A \to B$ et $\neg A \vee B$, entre $\neg(A \wedge B)$ et $\neg A \vee \neg B$, entre $\neg(A \vee B)$ et $\neg A \wedge \neg B$ et entre $\neg\neg A$ et A. Toute formule admet donc un représentant dans la classe de formules appartenant à la grammaire

$$A \ ::= \ X \mid \neg X \mid A \wedge A \mid A \vee A$$

tel que la formule soit classiquement prouvable si et seulement si son représentant est classiquement prouvable.

Mais on remarque que les règles d'introduction de la disjonction sont duales de celles de la conjonction. On peut donc se contenter d'un calcul où seul le côté droit du séquent est utilisé :

$$\frac{\vdash \Gamma}{\vdash \Gamma, A}\ \mathit{Aff} \qquad \frac{\vdash \Gamma, A, A}{\vdash \Gamma, A}\ \mathit{Cont} \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, B, A}\ \mathit{Ech}$$

$$\frac{}{\vdash \Gamma, A, A^{\perp}}\ \mathit{Ax} \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, A^{\perp}}{\vdash \Gamma, \Delta}\ \mathit{Coupe}$$

$$\frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \wedge B}\ \wedge_I \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, A \vee B}\ \vee_I^1 \qquad \frac{\vdash \Gamma, B}{\vdash \Gamma, A \vee B}\ \vee_I^2$$

où $^{\perp}$ est une opération involutive de négation définie par :

$$\begin{array}{rcl} X^{\perp} & = & \neg X \\ (\neg X)^{\perp} & = & X \\ (A \wedge B)^{\perp} & = & A^{\perp} \vee B^{\perp} \\ (A \vee B)^{\perp} & = & A^{\perp} \wedge B^{\perp} \end{array}$$

Nous qualifions de **monolatère** un tel calcul des séquents pour la logique classique, les calculs avec des formules de chaque côté du séquent étant qualifiés par opposition de **bilatère**.

Remarques : 1) Pour la règle de coupure du calcul des séquents monolatère, il y a deux formules de coupures duales l'une de l'autre. On parlera alors respectivement de l'occurrence conjonctive de la formule de coupure et de l'occurrence disjonctive de la formule de coupure.

2) Un tel calcul apparaît chez Schütte [53] en 1950 mais dans une formulation sans séquents, ceux-ci étant remplacés par la disjonction des formules les composant. Une version similaire est implicite dans des travaux de Novikoff [46] en 1941. Plus précisément, la logique que Novikoff étudie est une logique propositionnelle infinitaire. Il associe aux preuves de son calcul une notion de “régularité” qui revient à une notion de prouvabilité sans coupure dans un calcul des séquents monolatère pour lequel, là aussi, la notion de séquents est évitée par le recours à une disjonction. Un calcul des séquents monolatère pour la logique infinitaire classique avec une définition de séquents comparable à celle décrite ci-dessus peut être trouvé chez Tait [57]. Cependant, chez Tait, les séquents sont définis comme des ensembles de formules. Comme précisé en section 1.1.8, ceci implique un comportement calculatoire ambigu.

3- La proposition “faux”

Une manière alternative de considérer la négation est de la définir à partir de l'implication plutôt que de la prendre primitive. En revanche, ce que l'on prend alors comme primitif, c'est la proposition $\perp$ (proposition “fausse”) qui permet de définir $\neg A$ comme étant $A \to \perp$.

Les règles d'introduction de la négation deviennent (après oubli de la prémisse trivialement prouvable $\Gamma, \perp \vdash \perp$) :

$$\frac{\Gamma, A \vdash \Delta, \perp}{\Gamma \vdash \Delta, \neg A}\ \neg_d(=\to_d) \qquad \frac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta, \perp}\ \neg_g(=\to_g)$$

En adoptant ces règles, il ne peut plus apparaître de séquents sans formule à droite du symbole $\vdash$ dans une preuve. Dans le cas intuitionniste, cela veut dire qu'il y aura toujours une et une seule formule à droite du $\vdash$. Avec cette dernière condition, on se retrouve dans un calcul où la règle Aff_d ne peut plus être appliquée, et ce que l'on obtient est, en fait, une logique minimale (cf ci-dessous).

Pour récupérer la logique intuitionniste, il convient de considérer $\perp$ comme un connecteur 0-aire auquel est associé l'unique règle d'introduction

$$\frac{\Gamma, A \vdash B}{\Gamma, \perp \vdash B}\ \perp_g$$

Cette règle, considérée dans un calcul avec négation définie à partir de $\perp$, joue le même rôle que la règle Aff_d dans le calcul avec un connecteur primitif de négation. Cette règle est aussi appelée “ex falso quodlibet” (“du faux ce que l'on veut”).

4- La logique minimale

La logique **minimale** a été définie par Johansson [34] comme une logique intuitionniste sans “ex falso quodlibet”. Il n'y a pas de négation si ce n'est une négation formelle de la forme $A \rightarrow \bot$, mais avec une proposition $\bot$ à laquelle n'est associée aucune règle d'introduction.

Un calcul des séquents avec des séquents contenant exactement une formule à droite, et sans la règle $\bot_g$ constitue un calcul des séquents pour la logique minimale.

1.1.8 Remarques sur la définition de séquents

Plusieurs manières de définir les séquents (différentes de la définition originelle), existent dans la littérature. Elles ont toutes plus ou moins le projet d'éviter le recours à la règle d'échange. Cependant, les définitions des séquents comme ensembles ou multi-ensembles de formules ne permettent plus de distinguer entre certaines preuves qui, pourtant, se comportent différemment au niveau de l'élimination des coupures. Une conséquence directe de ceci est un non-déterminisme inévitable dans la définition du procédé l'élimination des coupures. Une réponse à ce problème est de considérer les séquents comme ensembles de formules nommées.

On ne considère ici que la version la plus générale de séquents, celle des séquents de la forme $\Gamma \vdash \Delta$ pour la logique classique, mais les remarques s'appliquent tout aussi bien aux cas de séquents pour les logiques intuitionniste et minimale.

1- Multi-ensemble ou ensemble de formules

Une première idée est de quotienter l'ordre, a priori non pertinent, des formules dans Γ et Δ. Cela revient à considérer Γ et Δ comme des multi-ensembles de formules, c'est-à-dire comme des ensembles de formules, chacune étant munie d'une arité. Une conséquence est que la règle d'échange n'a alors plus lieu d'être, ce que nous trouvons plutôt satisfaisant.

Une idée plus radicale est de considérer Γ et Δ non plus comme des multi-ensembles mais simplement comme des ensembles. Une conséquence est, qu'en plus de la règle d'échange, on économise aussi la règle de contraction. C'est l'option prise par exemple par Tait [57] que l'on trouvera reprise dans l'article de Schwichtenberg du “Handbook of mathematical logic” [54].

Ces définitions, valables au niveau de la prouvabilité, ne le sont plus au niveau calculatoire. En particulier, on ne peut plus définir l'élimination des coupures de manière non ambiguë. Schématiquement, c'est parce que des preuves, différentes sur le plan algorithme, sont identifiées à travers ces quotients.

Voici un exemple typique :

$$\dfrac{\dfrac{}{A \vdash A}\,Ax}{A, A \vdash A}\,\mathit{Aff}_g \qquad \text{et} \qquad \dfrac{\dfrac{\dfrac{}{A \vdash A}\,Ax}{A, A \vdash A}\,\mathit{Aff}_g}{A, A \vdash A}\,\mathit{Ech}_g$$

sont des preuves différentes sur un plan calculatoire (moralement l'une est associée au λ-terme $\lambda x.\lambda y.x$ et l'autre au λ-terme $\lambda x.\lambda y.y$) mais elles sont identifiées en considérant la suite A, A comme un multi-ensemble ou comme un ensemble.

2- Ensemble de formules nommées

Il existe une troisième alternative, celle de considérer des multi-ensembles avec différentiation des formules identiques. Cela revient en fait à considérer des ensembles de formules nommées : on associe à chaque formule de Γ un “nom” (ou un numéro) distinct du nom de toutes les autres formules de Γ, et, en particulier, différent du nom des autres occurrences de la même formule. Comme montré dans le chapitre 2, cette manière de définir les Γ a à voir avec les formulations du λ-calcul dans lesquelles les variables sont repérées par un nom. Elle est plus agréable à l'esprit, mais elle n'est pas exempte de complication lorsqu'on considère la “normalisation” des preuves. En particulier, on doit raisonner à α-conversion près.

Par opposition, la définition d'origine de Gentzen à base de suites de formules, comme cela est évoqué dans le chapitre 4, a à voir avec une formulation de type “avec indices de de Bruijn”.

Dans la suite, nous utiliserons la plupart du temps des ensembles de formules nommées pour définir les séquents. A cette fin, on admet l'existence de deux ensembles infinis $\mathcal{V}_H$ et $\mathcal{V}_C$ dont les éléments sont respectivement appelés **noms d'hypothèse** et **noms de conclusions**. Nous réservons les lettres x, y, ... pour noter les noms.

Une **formule nommée** consiste en la paire d'une formule et d'un nom. Le nom s'écrit en indice supérieur sous la forme A^x.

Un séquent a donc toujours la forme $\Gamma \vdash \Delta$ mais, avec cette nouvelle définition, Γ est un ensemble de formules nommées par des noms d'hypothèses et Δ est un ensemble de formules nommées par des noms de conclusions.

Dans la pratique, sauf pour souligner une ambiguïté, nous n'écrivons pas les noms des formules nommées. Aussi, une notation comme Δ, A désigne l'union de Δ et de $\{A\}$ où il est implicite que le nom de A n'interfère pas avec une formule A de même nom dans Δ.

1.1.9 Intégration des règles structurelles aux autres règles

Sans rien changer à l'agencement des règles d'introduction, d'axiome et de coupure entre elles, on peut agir sur la manières de traiter les règles structurelles. La manière de définir les séquents permet par exemple de se dispenser de la règle d'échange. On montre ici comment "camoufler" la règle d'affaiblissement et la règle de contraction.

1- Camouflage de la règle d'affaiblissement

Pour éviter de recourir à la règle d'affaiblissement, une possibilité est de repousser toutes les occurrences de règle d'affaiblissement vers les règles d'axiome, et de les y intégrer en modifiant la règle d'axiome de la sorte :

$$\frac{}{\Gamma, A \vdash \Delta, A}\ Ax$$

1- Camouflage de la règle de contraction

Pour éviter de recourir à la règle de contraction, une possibilité est de repousser toutes les occurrences de règle de contraction en direction des axiomes. Les règles d'introduction introduisant l'une des deux occurrences de la formule contractée font obstacle à cette remontée vers les axiomes. De même pour la règle de coupure.

Cependant, il est possible d'absorber les contractions au niveau des règles d'introduction en modifiant ces dernières de telle sorte qu'une occurrence de la formule introduite figure aussi dans les séquents conclusion des prémisses de la règle, tel l'exemple suivant :

$$\frac{\Gamma \vdash \Delta, A \wedge B, A \qquad \Gamma \vdash \Delta, A \wedge B, B}{\Gamma \vdash \Delta, A \wedge B}\ \wedge_d$$

On peut aussi absorber les contractions au niveau des règles de coupure en considérant la variante suivante de la règle de coupure :

$$\frac{\Gamma_1, \Delta \vdash \Delta_1, \Omega, A \qquad \Gamma_2, \Delta, A \vdash \Delta_2, \Omega}{\Gamma_1, \Gamma_2, \Delta \vdash \Delta_1, \Delta_2, \Omega}\ Coupe$$

1.2 Elimination des coupures

Se basant sur une approche "élimination des coupures = procédé de calcul", on peut décomposer les preuves du théorème d'élimination des coupures que l'on rencontre dans la littérature en deux parties : d'un côté la description d'un procédé de transformation de la preuve par une suite d'étapes élémentaires de transformation, de l'autre côté, une preuve que ce procédé débouche en un nombre d'étapes fini sur une preuve sans coupure. Cela s'applique aussi bien à la preuve originale du *Hauptsatz* par Gentzen, qu'aux autres variantes de cette preuve.

1.2.1 Procédures d'élimination des coupures

Un cadre général pour décrire l'élimination des coupures est celui des systèmes de règles de réécriture (cf par exemple Huet [32]).

Nous appellons **procédure d'élimination des coupures**, en abrégé **PEC**, tout système de règles de réécriture dont les preuves de la partie gauche des règles ont pour tête une coupure.

Nous appellons **rédex** toute preuve qui filtre la partie gauche d'une des règles de réécriture. On note $\xrightarrow{1}_R$ la notion de réduction engendré par congruence (fermeture compatible) à partir d'un système R de règles de réécriture et on note $\xrightarrow{*}_R$ la fermeture réflexive et transitive de $\xrightarrow{1}_R$.

Une preuve p est réductible selon R s'il existe au moins un r tel que $p \xrightarrow{1}_R r$.

Nous disons qu'une PEC est **exhaustive** si toute preuve ayant au moins une coupure est réductible. Une condition nécessaire et suffisante pour qu'une PEC soit exhaustive est que toutes les configurations de la forme

$$\dfrac{\dfrac{\vdots\, p_g}{\Gamma_1, \Delta \vdash \Delta_1, \Omega, A} \qquad \dfrac{\vdots\, p_d}{\Gamma_2, \Delta, A \vdash \Delta_2, \Omega}}{\Gamma_1, \Gamma_2, \Delta \vdash \Delta_1, \Delta_2, \Omega}\ Coupe$$

avec p_g et p_d sans coupures, soient prises en compte par au moins une des règles de réécriture.

Une procédure R est **confluente** si la notion de réduction $\xrightarrow{1}_R$ est confluente. En particulier si le système est orthogonal (sans paire critique et linéaire à gauche), il est confluent, ceci d'après le résultat de Huet [32] (cf section 2.4.3 pour une formulation plus précise de ce résultat).

Une preuve p est **normalisable** selon une certaine procédure R, s'il existe une preuve sans coupure r telle que $p \xrightarrow{*}_R r$. Une preuve p est **fortement normalisable** selon une procédure R dans les cas suivants :

- p est sans coupure
- pour tout r tel que $p \xrightarrow{1}_R r$, on a r fortement normalisable

Une procédure **termine** ou **termine faiblement** si toute preuve est normalisable selon cette procédure. Elle **termine fortement** si toute preuve est fortement normalisable

1.2.2 Ordres d'évaluation

Souhaitant appliquer une PEC à une preuve, il se pose le choix du rédex par lequel on commence à réduire.

On appelle **algorithme d'élimination des coupures** (en abrégé **AEC**) une PEC dont on a spécifié une stratégie de réduction (i.e. une fonction S associant à toute preuve p réductible un r tel que $p \xrightarrow{1}_R r$).

On dit qu'un AEC **termine** si pour tout p, il existe un entier n tel que $S^n(p)$ soit sans coupure.

Parmi les stratégies de réduction, deux types de stratégie offrent un intérêt particulier : les stratégies de réduction de la coupure la plus interne d'abord et les stratégie de réduction de la coupure la plus externe d'abord.

Les stratégies de réduction de la coupure la plus interne d'abord ont leur intérêt du fait qu'il est plutôt simple de prouver leur terminaison. C'est, dans la pratique, cette stratégie de réduction qui est implicitement considérée dans la plupart des preuves d'élimination des coupures (et en particulier dans la preuve originale de Gentzen).

Les stratégies de réduction de la coupure la plus externe d'abord ont leur intérêt du fait qu'elles correspondent à une interaction au sens de l'interprétation en termes de jeux décrite dans les chapitres 5 et 6. Les stratégies de réduction de la coupure la plus externe d'abord sont aussi des stratégies de réduction standard. Ainsi, dans le cas de preuves non bien fondées (par exemple des preuves interprétant l'opérateur de point fixe du λ-calcul), elles permettent d'atteindre le résultat, s'il existe, au bout d'un nombre d'étapes de réduction fini.

1.3 Elimination des coupures en logique classique

Souhaitant définir une procédure d'élimination des coupures en logique classique, on se retrouve confronté à divers choix dans la manière de réécrire les coupures et il est difficile de trouver de bons arguments permettant de trancher pour l'une ou l'autre des possibilités qui s'offrent.

Dans le but de mettre en évidence les différentes indéterminations qui apparaissent dans une procédure d'élimination des coupures en logique classique, nous présentons une preuve d'admissibilité des coupures de Martin-Löf dans [41].

Le calcul considéré par Martin-Löf est un calcul des séquents monolatère pour une logique propositionnelle infinitaire. Ici, nous nous restreignons au cas propositionnel fini. Les séquents de notre calcul sont considérés comme des ensembles de formules nommées, avec règles d'affaiblissement intégrées aux règles d'axiomes et avec règles de contraction intégrées aux règles d'introduction et de coupure.

Le faible nombre règles dans un tel calcul permet de simplifier la formulation de l'élimination des coupures tout en gardant l'essentiel du calcul et des problèmes inhérents à l'élimination des coupures.

Par commodité, on écrit le séquent $\vdash \Delta$ sous la simple forme Δ en oubliant le symbole $\vdash$.

Les formules du calcul des séquents monolatère pour la logique propositionnelle sont donc définies par la grammaire

$$A \quad ::= \quad X \mid \neg X \mid A \wedge A \mid A \vee A$$

et on reprend, pour l'opération involutive de négation $^\perp$, celle définie dans le paragraphe sur le calcul des séquents monolatère.

Les preuves sont construites à partir des règles d'inférence suivantes :

$$\frac{}{\Delta, A, A^\perp}\ Ax$$

$$\frac{\Delta, A \wedge B, A \quad \Delta, A \wedge B, B}{\Delta, A \wedge B}\ \wedge_I$$

$$\frac{\Delta, A \vee B, A}{\Delta, A \vee B}\ \vee_I^1 \qquad\qquad \frac{\Delta, A \vee B, B}{\Delta, A \vee B}\ \vee_I^2$$

$$\frac{\Gamma, \Omega, A \quad \Delta, \Omega, A^\perp}{\Gamma, \Delta, \Omega}\ Coupe$$

avec $A \wedge B$ pouvant aussi ne pas figurer dans les séquents conclusion des prémisses de la règle $\wedge_I$. Similairement pour $A \vee B$ dans les règles $\vee_I^1$ et $\vee_I^2$.

1.3.1 La preuve d'admissibilité de Martin-Löf

Nous considérons la preuve de Martin-Löf comme une preuve de l'éliminabilité d'une coupure unique. Nous décomposons sa preuve en une procédure d'élimination des coupures, un ordre de réduction pour cette stratégie, ce qui donne un AEC, puis une preuve de terminaison de cette AEC.

Nous profitons de l'occasion pour classifier les différentes règles de réécriture intervenant dans une procédure exhaustive. Si l'on considère les preuves situées de part et d'autre de la coupure, trois situations peuvent arriver :

1. l'inférence de tête introduit la formule de coupure

2. l'inférence de tête introduit une formule qui n'est pas la formule de coupure
3. l'inférence de tête est un axiome

Une analyse simultanée, sur la gauche et sur la droite de la coupure, du genre de règle d'inférence en tête conduit à considérer les cas qui suivent [1].

L'étape principale de réduction. L'étape de réduction que nous qualifions de principale, car elle correspond à une étape de calcul à proprement parler, est celle qui intervient lorsque des deux côtés de la coupure, la formule de coupure est introduite.

Si d'aucun côté de la coupure, la formule introduite ne figure dans les séquents conclusion des prémisses des règles d'introduction, cette étape se résoud ainsi :

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, A\end{array}}{\Delta, A \vee B}\vee_I^1 \qquad \dfrac{\begin{array}{c}\vdots\\ \Delta, A^\perp\end{array} \quad \begin{array}{c}\vdots\\ \Delta, B^\perp\end{array}}{\Delta, A^\perp \wedge B^\perp}\wedge_I}{\Delta}\,Coupe \quad \overset{r}{\to} \quad \dfrac{\begin{array}{c}\vdots\\ \Delta, A\end{array} \quad \begin{array}{c}\vdots\\ \Delta, A^\perp\end{array}}{\Delta}\,Coupe$$

Si la formule de coupure ne figure dans les séquents conclusion des prémisses que d'une seule des règles d'introduction de part et d'autre de la coupure, alors, il y a duplication de l'autre côté de la coupure. Exemple :

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, A\end{array}}{\Delta, A \vee B}\vee_I^1 \qquad \dfrac{\begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, A^\perp\end{array} \quad \begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, B^\perp\end{array}}{\Delta, A^\perp \wedge B^\perp}\wedge_I}{\Delta}\,Coupe \quad \overset{r}{\to} \quad \dfrac{\begin{array}{c}\vdots\\ \Delta, A\end{array} \quad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, A\end{array}}{\Delta, A \vee B}\vee_I^1 \quad \begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, A^\perp\end{array}}{\Delta, A^\perp}\,Coupe}{\Delta}\,Coupe$$

Si, en revanche, la formule de coupure figure dans les conclusions des prémisses de chacune des règles d'introduction de part et d'autre de la coupure, alors, il y a plusieurs manières de régler la situation. La solution choisie par Martin-Löf est répertoriée sous le nom concret de **coupures croisées** (originellement décrite par Gentzen dans [22]).

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, A \vee B, A\end{array}}{\Delta, A \vee B}\vee_I^1 \qquad \dfrac{\begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, A^\perp\end{array} \quad \begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, B^\perp\end{array}}{\Delta, A^\perp \wedge B^\perp}\wedge_I}{\Delta}\,Coupe$$

$$\overset{r}{\to} \quad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, A \vee B, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, A^\perp\end{array} \quad \begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, B^\perp\end{array}}{\Delta, A^\perp \wedge B^\perp}\wedge_I}{\Delta, A}\,Coupe \qquad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, A \vee B, A\end{array}}{\Delta, A \vee B}\vee_I^1 \quad \begin{array}{c}\vdots\\ \Delta, A^\perp \wedge B^\perp, A^\perp\end{array}}{\Delta, A^\perp}\,Coupe}{\Delta}\,Coupe$$

[1] Par simplicité, on ne représente qu'une seule partie partagée Δ du contexte de la coupure. Aussi, on considèrera qu'une contraction est intégrée aux règles d'introduction (et on fera alors figurer une occurence de la formule introduite dans la conclusion des prémisses de la règle d'introduction) que si cela a de l'importance pour la réduction.

Les étapes de commutation Lorsque d'un côté au moins de la coupure, on rencontre une inférence introduisant une formule qui n'est pas la formule de coupure, on peut faire passer cette inférence de l'autre côté de la coupure.

Ce cas peut arriver soit du côté où la formule de coupure est conjonctive, soit du côté où la formule de coupure est disjonctive (c'est-à-dire que, ci-dessous, A peut être conjonctive ou disjonctive) :

$$\dfrac{\dfrac{\overset{\vdots}{\Delta, C, A} \quad \overset{\vdots}{\Delta, D, A}}{\Delta, C \wedge D, A}\wedge_I \qquad \overset{\vdots}{\Delta, A^\perp}}{\Delta, C \wedge D}\;Coupe \quad \xrightarrow{r} \quad \dfrac{\dfrac{\overset{\vdots}{\Delta, C, A} \quad \overset{\vdots}{\Delta, A^\perp}}{\Delta, C}\;Coupe \qquad \dfrac{\overset{\vdots}{\Delta, D, A} \quad \overset{\vdots}{\Delta, A^\perp}}{\Delta, D}\;Coupe}{\Delta, C \wedge D}\wedge_I$$

et la même chose avec les règles d'introduction $\vee_I^1$ et $\vee_I^2$. Par exemple :

$$\dfrac{\overset{\vdots}{\Delta, A} \qquad \dfrac{\overset{\vdots}{\Delta, A^\perp, C}}{\Delta, A^\perp, C \vee D}\vee_I^1}{\Delta, C \vee D}\;Coupe \quad \xrightarrow{r} \quad \dfrac{\dfrac{\overset{\vdots}{\Delta, A} \quad \overset{\vdots}{\Delta, A^\perp, C}}{\Delta, C}\;Coupe}{\Delta, C \vee D}\vee_I^1$$

Les étapes axiomes Lorsque sur un des côtés de la coupure on rencontre un axiome mettant en jeu l'une des formules de la coupure, la coupure disparaît (là aussi, A peut être disjonctive ou conjonctive) :

$$\dfrac{\dfrac{}{\Delta, A^\perp, A}\;Ax \qquad \overset{\vdots}{\Delta, A^\perp}}{\Delta, A^\perp}\;Coupe \quad \xrightarrow{r} \quad \overset{\vdots}{\Delta, A^\perp}$$

Si les formules concernées par l'axiome ne sont pas impliquées dans la coupure, celle-ci disparaît aussi, mais différemment :

$$\dfrac{\overset{\vdots}{\Delta, A} \qquad \dfrac{}{\Delta, A^\perp, C^\perp, C}\;Ax}{\Delta, C^\perp, C}\;Coupe \quad \xrightarrow{r} \quad \dfrac{}{\Delta, C^\perp, C}\;Ax$$

Exhaustivité et non-confluence de cette procédure

L'ensemble des rédex définit une procédure d'élimination des coupures qui est clairement exhaustive. Mais le système de règles de réécriture contient des paires critiques non confluentes. Les paires critiques correspondent aux cas où des deux côtés de la coupure, la règle de tête n'introduit pas la formule de coupure.

$$\dfrac{\dfrac{\overset{\vdots}{\Delta, C, A} \quad \overset{\vdots}{\Delta, D, A}}{\Delta, C \wedge D, A}\wedge_I \qquad \dfrac{\overset{\vdots}{\Delta, A^\perp, E} \quad \overset{\vdots}{\Delta, A^\perp, F}}{\Delta, A^\perp, E \wedge F}\wedge_I}{\Delta, C \wedge D, E \wedge F}\;Coupe$$

et

$$\dfrac{\dfrac{}{\Delta, A, C^\perp, C}\,Ax \quad \dfrac{}{\Delta, A^\perp, D^\perp, D}\,Ax}{\Delta, C^\perp, C, D^\perp, D}\,Coupe$$

Dans le premier cas, on peut aussi rencontrer les règles $\vee_I^1$ et $\vee_I^2$ à la place de l'une ou des deux règles $\wedge_I$.

Il faut aussi considérer les cas mixtes où, d'un côté, il y a une règle d'introduction introduisant une formule autre que la formule de coupure et, de l'autre côté, un axiome ne mettant pas non plus en jeu la formule de coupure.

Néanmoins il est très simple de restreindre la procédure en une procédure confluente en spécialisant les règles de réécriture pour lesquelles, sur la gauche (par exemple), il y a une régle d'introduction ou un axiome ne concernant pas la formule de coupure, en une règle où, sur la droite, on n'autorise que les règles d'introduction de la formule de coupure.

Remarque : Les paires critiques impliquant un axiome concernant une formule de coupure sont elles convergentes.

Preuve de terminaison On considère une coupure entre deux preuves sans coupures. La preuve d'éliminabilité de cette coupure est par induction (structurelle) d'abord sur la formule de coupure puis sur la paire des preuves de part et d'autre de la coupure.

Il suffit alors d'observer que chacune des règles de réécriture transforme une coupure sur la formule A et entre les preuves sans coupures p et q en une configuration pour laquelle, compte tenu de la stratégie de réduction employée, les nouvelles coupures sont telles que soit l'une des preuves à droite ou à gauche est plus petite (avec une même formule de coupure), soit que la formule de coupure est plus petite.

Ceci revient à prouver la terminaison de la procédure évaluée selon une stratégie de réduction de la coupure la plus interne d'abord.

1.3.2 L'étape principale de réduction

C'est le cas où des deux côtés de la coupure, on introduit la formule de coupure.

Si d'aucun côté de la coupure, il n'y a de contraction intégrée aux règles d'introduction, ou s'il n'y a une contraction intégrée que d'un seul côté de la coupure, alors, on ne connait qu'une seule manière de réduire la coupure. C'est ce qui arrive systématiquement en logique intuitionniste.

En revanche, et c'est là un trait propre à la logique classique, si des deux côtés de la coupure, il y a une contraction intégrée aux règles d'introduction (c'est-à-dire qu'une occurrence de la formule introduite figure aussi dans les conclusions des prémisses des règles d'introduction), alors on rencontre dans la littérature trois manières de réduire cette configuration.

On considère la coupure suivante :

$$\dfrac{\begin{array}{c}\vdots\ \rho\\ \Omega, \Delta, A \vee B\end{array} \quad \begin{array}{c}\vdots\ \sigma\\ \Omega, \Gamma, A^\perp \wedge B^\perp\end{array}}{\Omega, \Gamma, \Delta}\,Coupe$$

entre

$$\rho = \dfrac{\begin{array}{c}\vdots\ \rho_1\\ \Omega, \Delta, A \vee B, A\end{array}}{\Omega, \Delta, A \vee B}\,\vee_I^1 \qquad \text{et} \qquad \sigma = \dfrac{\begin{array}{c}\vdots\ \sigma_1\\ \Omega, \Gamma, A^\perp \wedge B^\perp, A^\perp\end{array} \quad \begin{array}{c}\vdots\ \sigma_2\\ \Omega, \Gamma, A^\perp \wedge B^\perp, B^\perp\end{array}}{\Omega, \Gamma, A^\perp \wedge B^\perp}\,\wedge_I$$

On peut réécrire ainsi :

Conjonction dupliquante

Dans un premier temps, la prémisse portant l'occurrence conjonctive de la formule de coupure duplique l'autre prémisse de la coupure et distribue deux de ces copies sur ses propres prémisses :

$$\dfrac{\overset{\vdots\,\rho}{\Omega,\Delta,A\vee B} \qquad \overset{\vdots\,\sigma'}{\Omega,\Gamma,\Delta,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta}\ Coupe$$

avec

$$\sigma' = \dfrac{\dfrac{\overset{\vdots\,\rho}{\Omega,\Delta,A\vee B} \qquad \overset{\vdots\,\sigma_1}{\Omega,\Gamma,A^\perp\wedge B^\perp,A^\perp}}{\Omega,\Gamma,\Delta,A^\perp}\ Coupe \qquad \dfrac{\overset{\vdots\,\rho}{\Omega,\Delta,A\vee B} \qquad \overset{\vdots\,\sigma_2}{\Omega,\Gamma,A^\perp\wedge B^\perp,B^\perp}}{\Omega,\Gamma,\Delta,B^\perp}\ Coupe}{\Omega,\Gamma,\Delta,A^\perp\wedge B^\perp}\ \wedge_I$$

Ce n'est qu'après que ρ duplique à son tour la preuve avec laquelle elle est coupée :

$$\dfrac{\dfrac{\dfrac{\overset{\vdots\,\rho_1}{\Omega,\Delta,A\vee B,A} \qquad \overset{\vdots\,\sigma'}{\Omega,\Gamma,\Delta,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta,A}\ Coupe}{\Omega,\Gamma,\Delta,A\vee B}\ \vee_I^1 \qquad \overset{\vdots\,\sigma'}{\Omega,\Gamma,\Delta,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta}\ Coupe$$

Après simplification de la coupure la plus externe, qui s'avère être une coupure sans contraction ni à droite ni à gauche, on obtient cette configuration.

$$\dfrac{\dfrac{\overset{\vdots\,\rho_1}{\Omega,\Delta,A\vee B,A} \qquad \overset{\vdots\,\sigma'}{\Omega,\Gamma,\Delta,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta,A}\ Coupe \qquad \dfrac{\overset{\vdots\,\rho}{\Omega,\Delta,A\vee B} \qquad \overset{\vdots\,\sigma_1}{\Omega,\Gamma,A^\perp\wedge B^\perp,A^\perp}}{\Omega,\Gamma,\Delta,A^\perp}\ Coupe}{\Omega,\Gamma,\Delta}\ Coupe$$

Disjonction dupliquante

Lorsque c'est ρ qui duplique d'abord σ, en suivant le même procédé dual du précédemment, on obtient :

$$\dfrac{\dfrac{\overset{\vdots\,\rho_1}{\Omega,\Delta,A\vee B,A} \qquad \overset{\vdots\,\sigma}{\Omega,\Gamma,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta,A}\ Coupe \qquad \dfrac{\overset{\vdots\,\rho'}{\Omega,\Gamma,\Delta,A\vee B} \qquad \overset{\vdots\,\sigma_1}{\Omega,\Gamma,A^\perp\wedge B^\perp,A^\perp}}{\Omega,\Gamma,\Delta,A^\perp}\ Coupe}{\Omega,\Gamma,\Delta}\ Coupe$$

avec

$$\rho' = \dfrac{\dfrac{\overset{\vdots\,\rho_1}{\Omega,\Delta,A\vee B,A} \qquad \overset{\vdots\,\sigma}{\Omega,\Gamma,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta,A}\ Coupe}{\Omega,\Gamma,\Delta,A\vee B}\ \vee_I^1$$

Coupures croisées

La solution des coupures croisées apparaît comme une version simplificatrice et symétrisante de chacune des deux solutions précédentes. De fait, on simplifie en remarquant que les "gros morceaux" du milieu, σ' ou ρ', peuvent être obtenus directement en récupérant des bouts de preuves plus simples déjà connus.

$$\dfrac{\dfrac{\dfrac{\vdots\ \rho_1}{\Omega,\Delta,A\vee B,A} \quad \dfrac{\vdots\ \sigma}{\Omega,\Gamma,A^\perp\wedge B^\perp}}{\Omega,\Gamma,\Delta,A}\,Coupe \qquad \dfrac{\dfrac{\vdots\ \rho}{\Omega,\Delta,A\vee B} \quad \dfrac{\vdots\ \sigma_1}{\Omega,\Gamma,A^\perp\wedge B^\perp,A^\perp}}{\Omega,\Gamma,\Delta,A^\perp}\,Coupe}{\Omega,\Gamma,\Delta}\,Coupe$$

Autres protocoles ?

Chacune de ces trois règles propose donc une manière de gérer une interaction entre deux formules qui peuvent être, chacune, introduite plusieurs fois. Ces trois règles sont les plus simples parmi celles que l'on peut imaginer. Toutefois, si on prend en compte les possibles triples contractions, ou même quadruples contractions, de formules, et qu'on cherche à réécrire tout ce beau monde en de multiples coupures mais toutes de taille plus petite, il s'ouvre devant nous un univers purement combinatoire d'association des preuves entre elles.

A l'opposé, soulignons que, a priori, seules les règles avec conjonction dupliquante et disjonction dupliquante sont compatibles avec une interprétation des preuves à l'aide, par exemple, d'espaces cohérents, et en particulier, avec une interprétation à l'aide de λ-termes.

Notons aussi que nous avons distingué entre conjonction dupliquante et disjonction dupliquante alors que chez la plupart des auteurs, tant que l'aspect calculatoire de la preuve n'est pas considéré, c'est une latéralité gauche ou droite dépendant arbitrairement de la façon d'écrire la preuve sur la feuille qui décide de la règle de réécriture à appliquer (c'est par exemple le cas dans la preuve d'origine de Gentzen).

Remarque : C'est l'étape principale de réduction qui contient la partie la plus complexe de l'élimination des coupures (l'analogue en λ-calcul est la règle $(\lambda x.u\ v) \to u[x := v]$). En comparaison, les autres étapes de réduction s'apparentent à des étapes de propagation de substitution (cf à ce sujet le chapitre 2).

1.3.3 Sur le non-déterminisme et la confluence

Il était (et il est encore parfois) d'usage de considérer l'élimination des coupures en logique classique comme fondamentalement non-déterministe. A l'appui de ce point de vue, il y a les paires critiques montrées en section 1.3.1 pour lesquelles il n'y a, a priori, aucune raison de choisir l'une ou l'autre des manières de réduire. Ce n'est peut-être pas un mauvais point de vue, surtout si on entend bien par non-déterminisme, l'idée d'une procédure qui réduit de plusieurs manières différentes en parallèle et qui fournit un ensemble de résultats finaux plutôt qu'un seul résultat. Le problème de cette approche est la question de comment aborder le non-déterminisme de manière théorique, et de quoi affirmer d'intéressant sur ce non-déterminisme.

Si l'on se cantonne à une élimination séquentielle des coupures, le non-déterminisme devient non confluence. La non confluence n'a rien d'irremédiable en soi : il suffit pour faire d'une procédure non confluente une procédure confluente, d'interdire certaines réductions. Le seul problème de ce procédé de "confluentisation" est de justifier pourquoi ce serait l'une plutôt que l'autre de ces réductions génératrices de paires critiques qu'il faudrait interdire.

Un argument important pour éliminer l'arbitraire du procédé de confluentisation repose sur les "rôles" respectifs des connecteurs et quantificateurs. De fait, tout énoncé est prouvable sans recourir à des contractions sur les conjonctions, sur les disjonctions, et sur les quantifications universelles. En revanche, on ne peut éviter, dans le cas général, les contractions sur les quantifications existentielles. Cela met en évidence une différence "fondamentale" entre le rôle de la quantification existentielle et le rôle de la quantification universelle (cette distinction est moins claire dans le cas purement propositionnel). L'un "émet", l'autre "reçoit".

Un autre argument pour éliminer l'arbitraire de la confluentisation est de ne garder des différents comportements calculatoires possibles de l'élimination des coupures, que ceux qui ont un sens dans le cadre d'une interprétation à l'aide de λ-termes. On qualifiera informellement de "substitutives" de telles procédures.

Toujours est-il que pour ces paires critiques citées en section 1.3.1 et qui sont de fait l'exemple typique du genre de problème que l'on rencontre pour l'élimination des coupures en logique classique, il y a deux manières de trancher, que nous appellerons maîtrise conjonctive et maîtrise disjonctive.

Maîtrise conjonctive

On ne garde des 2 réductions possibles pour l'étape de commutation que celle-ci :

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, C, A\end{array} \quad \begin{array}{c}\vdots\\ \Delta, D, A\end{array}}{\Delta, C \wedge D, A}\wedge_I \quad \begin{array}{c}\vdots\\ \Delta, A^{\perp}, E\end{array}}{\Delta, C \wedge D, E}\,Coupe \qquad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, C, A\end{array} \quad \begin{array}{c}\vdots\\ \Delta, D, A\end{array}}{\Delta, C \wedge D, A}\wedge_I \quad \begin{array}{c}\vdots\\ \Delta, A^{\perp}, F\end{array}}{\Delta, C \wedge D, F}\,Coupe}{\Delta, C \wedge D, E \wedge F}\wedge_I$$

On ne garde des 2 réductions possibles pour l'étape axiomes que celle-ci :

$$\dfrac{}{\Delta, D^{\perp}, D, \mathbf{C}^{\perp}, \mathbf{C}}\,Ax$$

Maîtrise disjonctive

On ne garde des 2 réductions possibles pour l'étape de commutation que celle-ci :

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Delta, C, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \Delta, A^{\perp}, E\end{array} \quad \begin{array}{c}\vdots\\ \Delta, A^{\perp}, F\end{array}}{\Delta, A^{\perp}, E \wedge F}\wedge_I}{\Delta, E \wedge F, C}\,Coupe \qquad \dfrac{\begin{array}{c}\vdots\\ \Delta, D, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \Delta, A^{\perp}, E\end{array} \quad \begin{array}{c}\vdots\\ \Delta, A^{\perp}, F\end{array}}{\Delta, A^{\perp}, E \wedge F}\wedge_I}{\Delta, E \wedge F, D}\,Coupe}{\Delta, C \wedge D, E \wedge F}\wedge_I$$

On ne garde des 2 réductions possibles pour l'étape axiomes que celle-ci :

$$\dfrac{}{\Delta, C^{\perp}, C, \mathbf{D}^{\perp}, \mathbf{D}}\,Ax$$

1.3.4 Un panorama des procédures existantes

On donne très brièvement un aperçu de quelques preuves d'élimination des coupures trouvables dans la littérature.

La procédure originelle de Gentzen

La procédure de Gentzen est une procédure réduite selon un algorithme de réduction de la coupure la plus interne d'abord.

Pour les étapes principales de réduction, c'est la place de la preuve portant l'occurrence conjonctive de la formule de coupure, à gauche ou à droite de la représentation de la coupure sur la feuille de papier qui importe. Dans le premier cas, la conjonction est dupliquante, dans le second cas c'est la disjonction qui duplique d'abord (si bien qu'on pourrait parler, pour simplifier, de côté gauche dupliquant).

Pour les conflits lors des étapes de commutation et axiome, le même critère est pris en compte : la position, à gauche ou à droite, de l'occurrence conjonctive de la formule de coupure.

Terminaison forte de la procédure originelle de Gentzen

Une première preuve de terminaison forte et non-déterminisite de la procédure de Gentzen est dû à Dragalin [15] en 1979.

Plus récemment, une autre preuve a été donnée par Tahhan [56] pour une restriction confluente, et au cas propositionnel, de la procédure de Gentzen. Sa preuve repose sur le résultat de réécriture suivant de O'Donnell [47] :

Si un système de réécriture orthogonal termine en réduisant systématiquement des rédex les plus internes, alors il termine fortement.

La permutation des coupures et la procédure de Girard

Une procédure telle que celle implicite à la preuve d'admissibilité de Martin-Löf a la propriété que les paires critiques s'appliquent à la même coupure. Ce n'est pas le cas de la procédure de Girard dans [27] ou [24].

L'interaction à la Coquand dans le calcul de Novikoff

On décrit au chapitre 6 une manière d'éliminer les coupures en passant par une interprétation de la prouvabilité en termes de jeux. Cette "interaction" entre preuves, définie par Coquand dans [9], si on la considère en oubliant le vocabulaire de la théorie des jeux s'exprime comme une PEC réduite selon une stratégie de réduction de la coupure la plus externe d'abord (cf chapitre 6). Cette PEC est confluente et dans tous les cas de réduction, c'est la conjonction qui duplique et a la maîtrise.

La solution LC

Le calcul LC, décrit dans Girard [25], est un calcul où toutes les formules reçoivent une polarité positive ou négative. La PEC n'est pas très précisément décrite dans l'article. Ce que l'on peut on dire, c'est qu'elle est confluente et que l'étape principale de réduction dépend de la polarité. La conjonction duplique et a la maîtrise si l'occurrence conjonctive de la formule de coupure est négative, sinon, c'est la disjonction.

Solution Coquand = solution LC pour le fragment $\forall/\exists$

Puisque la quantification universelle est de polarité négative dans LC, pour le fragment $\forall/\exists$ de LC, comme pour la procédure associée à l'interaction de Coquand, c'est la conjonction qui duplique et a la maîtrise.

Un travail peu connu : l'approche de Novikoff

Novikoff montre dans [46] que toute preuve est – ce qu'il appelle – régulière. Sa notion de régularité revient à une notion de preuve sans coupure en logique propositionnelle infinitaire et il décrit au passage un procédé délimination des coupures.

La solution du lemme d'inversion

Dans Tait [57] (repris par Schwichtenberg [54]) on trouve une procédure qui ne pose le problème du choix de l'étape principale de réduction pour la raison que les contractions sur les formules conjonctives sont évités avant de faire la coupure, et ceci à l'aide d'un procédé de déplacement de ces contractions vers les sous-formules disjonctives les plus proches.

Dans le cas général, cette procédure ne se comporte comme aucune des PEC évoquées ci-dessus.

PEC interprétables en logique lineaire

Les procédures pour LC, de Coquand pour le calcul de Novikoff, pour la déduction libre et pour certaines versions déterministes du λ-calcul symétrique sont interprétables par la procédure d'élimination des coupures de la version calcul des séquents de la logique linéaire.

Réductions non "substitutives"

Toute procédure ayant recours aux coupures croisées n'est, a priori, pas interprétable dans la logique linéaire.

C'est le cas aussi de la procédure de Girard dans [27] et des procédures basées sur un lemme d'inversion de la conjonction (comme dans Tait [57] ou Schwichtenberg [54]).

C'est aussi le cas de la solution du "et symétrique" décrite dans Coquand *et al* [10].

Calcul des séquents et λ-calcul

Chapitre 2

LJT et le $\overline{\lambda}$-calcul

Il existe une interprétation standard des preuves sans coupure de LJ par des λ-termes simplement typés normaux. Cette interprétation a été décrite par D. Prawitz [50] et, par exemple, par J. Zucker [60] et G. Pottinger [49].

Cette interprétation n'est pas une correspondance dans le sens où plusieurs preuves de LJ sont associées au même λ-terme simplement typé. Nous décrivons alors une restriction de LJ, que nous appelons LJT, pour laquelle il y a effectivement correspondance bijective entre les preuves sans coupures et les λ-termes simplement typés normaux.

Que se passe-t-il au niveau de la correspondance si l'on considère des preuves avec coupures et des λ-termes non normaux ? Si l'on s'intéresse aux étapes élémentaires du calcul explicite de la substitution dans le λ-calcul, on n'a pas de correspondance avec les étapes élémentaires d'élimination des coupures telles que caractérisées dans les chapitres précédents. Ainsi, pour obtenir une correspondance à la fois au niveau des preuves et des termes mais aussi au niveau de la réduction, bref, pour avoir un isomorphisme entre LJT et le λ-calcul simplement typé, nous sommes amené à considérer une syntaxe alternative pour "le" λ-calcul qui, elle, est isomorphe à la structure du calcul des séquents. Nous appelons $\overline{\lambda}$-calcul cette autre syntaxe.

Nous ne considérons dans ce chapitre que le fragment de LJ avec pour seul connecteur l'implication. Après une étude de l'interprétation de LJ par des λ-termes, nous introduisons le calcul LJT, puis, le $\overline{\lambda}$-calcul (dont les éléments sont appelés $\overline{\lambda}$-expressions), avant de caractériser l'isomorphisme et de prouver la confluence et la forte terminaison de la réduction des $\overline{\lambda}$-expressions simplement typées.

L'extension de l'isomorphisme à la logique classique, ainsi qu'aux connecteurs autres que l'implication, est présentée dans les chapitres suivants.

2.1 Correspondance entre les preuves sans coupures de LJ et les λ-termes simplement typés normaux

Nous donnons une interprétation des preuves du fragment minimal de LJ par des λ-termes, dans le style de ce qui est suggéré par Prawitz [50]. Cette interprétation est aussi décrite, avec plus de précision, dans Zucker [60], Pottinger [49] ou Mints [43].

On considère une version sans coupure de LJ avec l'implication comme seul connecteur (le cas des autres connecteurs est traités dans le chapitre 4). On adopte une règle explicite de contraction mais la règle d'affaiblissement est intégrée à la règle d'axiome. La partie gauche des séquents consiste en une suite de formules nommées. On choisit, pour ensemble de noms, l'ensemble $\mathcal{V}$ des noms de variables du λ-calcul et on note les formules nommées sous la forme $x:A$ plutôt que sous la forme A^x (comme cela est fait, par exemple, dans le chapitre précédent).

Les formules sont définies par la grammaire

$$A \ ::= \ X \mid A \to A$$

où X parcourt l'ensemble $\mathcal{V}_F$ des variables propositionnelles.

Les preuves sont contruites à partir des règles suivantes :

$$\frac{}{\Gamma, A \vdash A}\ Ax \qquad \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ Cont$$

$$\frac{\Gamma \vdash A \quad \Gamma, B \vdash C}{\Gamma, A \to B \vdash C}\ \to_g \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ \to_d$$

L'interprétation de Prawitz procède inductivement comme suit :

$$\frac{}{\Gamma, x:A \vdash x:A}\ Ax \qquad \frac{\Gamma, x:A, y:A \vdash u:B}{\Gamma, z:A \vdash u\{y := z\}\{x := z\}:B}\ Cont$$

$$\frac{\Gamma \vdash u:A \quad \Gamma, y:B \vdash v:C}{\Gamma, x:A \to B \vdash v\{y := (x\ u)\}:C}\ \to_g \qquad \frac{\Gamma, x:A \vdash u:B}{\Gamma \vdash \lambda x.u:A \to B}\ \to_d$$

où $v\{x := u\}$ représente le terme v dans lequel chaque occurence de x a été remplacée par le terme u.

Un exemple de preuves associées au même λ-terme :

$$\dfrac{\dfrac{\dfrac{}{A \to B, A, C \vdash A}\,Ax \quad \dfrac{}{A \to B, A, C, B \vdash B}\,Ax}{A \to B, A, C \vdash B}\,\to_g}{A \to B, A \vdash C \to B}\,\to_d$$

et

$$\dfrac{\dfrac{}{A \to B, A \vdash A}\,Ax \quad \dfrac{\dfrac{}{A \to B, A, B, C \vdash B}\,Ax}{A \to B, A, B \vdash C \to B}\,\to_d}{A \to B, A \vdash C \to B}\,\to_g$$

sont toutes deux associées au λ-terme $\lambda z.(x\ y) : C \to B$ pour lequel $z : C$, $x : A \to B$ et $y : A$.

2.2 Vers le calcul LJT

LJT s'obtient à partir de LJ en restreignant l'utilisation de la règle $\to_g$. On n'autorise à voir sur le côté droit d'une instance de la règle $\dfrac{\Gamma \vdash A \quad \Gamma, B \vdash C}{\Gamma, A \to B, A \vdash C \to B}\,\to_g$ qu'un axiome déclarant B ou bien une autre règle $\to_g$ introduisant B (dans ce cas B est donc de la forme $D \to E$).

Cette restriction est exprimée par l'utilisation d'un "bénitier" à la manière de J.-Y. Girard dans [25, 26]. Ainsi, il y a deux types de séquents dans le calcul LJT, ceux ayant la forme $\Gamma; \vdash A$ et ceux ayant la forme $\Gamma; A \vdash B$. Une notation synthétique pour les séquents est celle-ci : $\Gamma; \Pi \vdash A$. Ce qu'on appelle **bénitier** est la place se situant juste à droite du ";". Le bénitier peut être vide ou plein auquel cas il contient exactement une formule (non nommée). La notation Π désigne un tel ensemble qui est soit vide soit restreint à une unique formule.

Le fragment sans coupure de LJT

règle d'axiome	règle de déchargement/contraction
$\dfrac{}{\Gamma; A \vdash A}\,Ax$	$\dfrac{\Gamma, A; A \vdash B}{\Gamma, A; \vdash B}\,Déch$
règle d'introduction gauche de $\to$	règle d'introduction droite de $\to$
$\dfrac{\Gamma; \vdash A \quad \Gamma; B \vdash C}{\Gamma; A \to B \vdash C}\,\to_g$	$\dfrac{\Gamma, A; \vdash B}{\Gamma; \vdash A \to B}\,\to_d$

Le calcul LJT a donc la propriété que ses preuves sans coupures sont en bijection avec les λ-termes simplement typés normaux.

Considérons par exemple un terme normal de la forme $\lambda x_1 ... \lambda x_n.(y\ u_1\ ... u_p)$, possèdant un type de la forme $A_1 \to ... \to A_n \to B$ et dans lequel y bénéficie d'un type $D = C_1 \to ... \to C_p \to B$, et qui, lui-même, lui associe canoniquement la preuve

$$\dfrac{\Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}; \vdash u_p\!:\!C_p \qquad \dfrac{}{\Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}; y\!:\!B \vdash y\!:\!B}\,Ax}{\Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}; y_n\!:\!C_p \to B \vdash (y_n\ u_n)\!:\!B}\,\to_g$$

$$\vdots$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}; \vdash u_1\!:\!C_1 \qquad \Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}; y_2\!:\!C_2 \to \ldots \to C_p \to B \vdash (y_2\ u_2\ \ldots u_n)\!:\!B}{\Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}\ ; y_1\!:\!D \vdash\ (y_1\ u_1\ \ldots u_n)\!:\!B}\,\to_g}{\Gamma, y\!:\!D, \overrightarrow{x_{1..n}\!:\!A_{1..n}}\ ;\ \vdash\ (y_1\ u_1\ \ldots u_n)\!:\!B}\,\textit{Déch}}{\Gamma, y\!:\!D, \overrightarrow{x_{1..n-1}\!:\!A_{1..n-1}};\ \vdash\ \lambda x_n.(y_1\ u_1\ \ldots u_n)\!:\!A_n \to B}\,\to_d}{}$$

$$\vdots$$

$$\dfrac{\Gamma, y\!:\!D, x_1\!:\!A_1\ ;\ \vdash \lambda x_2 \ldots \lambda x_n.(y_1\ u_1\ \ldots u_n)\!:\!A_2 \to \ldots \to A_n \to B}{\Gamma, y\!:\!D\ ;\ \vdash \lambda x_1 \ldots \lambda x_n.(y_1\ u_1\ \ldots u_n)\!:\!A_1 \to \ldots \to A_n \to B}\,\to_d$$

Remarque : Le calcul LJT a déjà été considéré par Danos, Joinet et Schellinx [14], avec une manière différente de gérer les règles structurelles. Chez Danos *et al*, LJT apparaît comme le fragment intuitionniste de LKT, une restriction du calcul LK ayant la propriété de se plonger simplement dans la logique linéaire : les preuves de LKT sont "décorables", c'est-à-dire, par l'insertion adéquate de "?" et de "!" dans les formules, les preuves de LKT s'interprètent comme des preuves de la logique linéaire.

Le calcul LJT est aussi un fragment strict du fragment neutre intuitionniste de la logique unifiée décrite par Girard [26]. Ce calcul de Girard est lui-même une version contrainte par un "bénitier" de LJ. Girard remarquait à ce sujet que "la formule [dans le bénitier] (s'il y en a une) est l'analogue de la familière *variable de tête* pour le λ-calcul typé". Nous verrons aussi plus loin comment retrouver LJ dans LJT.

Il nous faut aussi signaler qu'un tel calcul apparaît dans l'article de Howard [31] sur l'interprétation de la déduction naturelle comme λ-calcul typé. Bien que traitant plutôt de la déduction naturelle, on trouve dans cet article une formulation d'un calcul des séquents comparable à LJT et dont il est dit que ses preuves sont en bijection avec les λ-termes normaux simplement typés.

Les règles de coupures dans LJT

Il y a deux sortes de règles de coupures dans LJT, selon que la formule de coupure est ou non dans le bénitier (chacune d'entre elles se divisant en deux selon que Π est vide ou non) :

règle de coupure de tête — règle de coupure médiane

$$\dfrac{\Gamma;\Pi \vdash A \quad \Gamma;A \vdash B}{\Gamma;\Pi \vdash B}\,Coup_T \qquad \dfrac{\Gamma; \vdash A \quad \Gamma, A;\Pi \vdash B}{\Gamma;\Pi \vdash B}\,Coup_M$$

Il s'avère que la règle de coupure médiane s'interprète naturellement comme un opérateur *explicite* de substitution :

$$\dfrac{\Gamma; \vdash v\!:\!A \quad \Gamma, x\!:\!A;\Pi \vdash u\!:\!B}{\Gamma;\Pi \vdash u[x := v]\!:\!B}\,Coup_M$$

$u[x := v]$ est une telle notation pour une opération, intégrée dans la syntaxe, de substitution de la variable de nom x par le terme v dans le terme u.

La question de l'élimination des coupures en LJT

On peut décrire une procédure d'élimination des coupures de tête et médiane à l'aide d'un système de réécriture.

En particulier, on a la règle de réécriture suivante :

$$\dfrac{\Gamma;\vdash v:A \qquad \dfrac{\Gamma, x:A;\vdash u_1 : A_1 \quad \Gamma, x:A; y:A_2 \to \ldots \to A_n \to B \vdash (y\ u_2\ \ldots\ u_n):B}{\Gamma, x:A; y:A_1 \to \ldots \to A_n \to B \vdash (y\ u_1\ \ldots\ u_n):B}\to_g}{\Gamma; y:A_1 \to \ldots \to A_n \to B \vdash (y\ u_1\ \ldots\ u_n)[x:=v]:B}\mathit{Coup}_M$$

se réécrit en

$$\dfrac{\dfrac{\Gamma;\vdash v:A \quad \Gamma, x:A;\vdash u_1 : A_1}{\Gamma;\vdash u_1[x:=v]:A_1}\mathit{Coup}_M \qquad \dfrac{\Gamma;\vdash v:A \quad \Gamma, x:A; y:A_2 \to \ldots \to A_n \to B \vdash (y\ u_2\ \ldots\ u_n):B}{\Gamma; y:A_2 \to \ldots \to A_n \to B \vdash (y\ u_2\ \ldots\ u_n)[x:=v]:B}\mathit{Coup}_M}{\Gamma; y:A_1 \to \ldots \to A_n \to B \vdash (y\ u_1[x:=v]\ \ldots\ u_n)[x:=v]:B}\to_g$$

Cependant cette règle de réduction n'a pas de contrepartie en λ-calcul. Effectivement, au niveau des λ-termes, elle correspond à la réduction d'un terme de la forme $(y\ u_1\ \ldots\ u_n)[x:=v]$ en le terme $(y\ u_1[x:=v]\ \ldots\ u_n)[x:=v]$, tandis que le résultat attendu de la réduction aurait été $((y\ u_1\ \ldots u_{n-1})[x:=v]\ u_n[x:=v])$.

Cette différence provient de la manière de construire les termes de la forme $(y\ u_1\ \ldots\ u_n)$ dans LJT : au lieu de construire d'abord $(y\ u_1)$ puis $(y\ u_1\ u_2)$, ... on construit d'abord $(y\ u_n)$ puis $(y\ u_{n-1}\ u_n)$ et enfin $(y\ u_1\ \ldots\ u_n)$. Cette différence est la motivation pour introduire le $\overline{\lambda}$-calcul : au lieu de considérer les applications sous la forme $((u\ u_1)\ldots u_n)$, on va les considérer sous la forme $(u\ [u_1;\ldots;u_n])$, c'est-à-dire comme l'application – binaire – d'une fonction à la liste de ses arguments.

2.3 Le plongement de LJ dans LJT

LJT a le même pouvoir que LJ en terme de prouvabilité. De fait, chacune des règles de LJ peut s'exprimer comme un assemblage de règles de LJT dont la conclusion est de bénitier vide.

Pour exprimer la traduction, on préfére une version de LJ où la contraction est intégrée à la règle $\to_g$. La traduction est la suivante :

$$\dfrac{}{\Gamma, A \vdash A}Ax \quad \text{se traduit en} \quad \dfrac{\dfrac{}{\Gamma, A; A \vdash A}Ax}{\Gamma, A;\vdash A}\mathit{Déch}$$

$$\dfrac{\begin{matrix}\vdots\\ \Gamma, A \to B \vdash A\end{matrix} \quad \begin{matrix}\vdots\\ \Gamma, A \to B, B \vdash C\end{matrix}}{\Gamma, A \to B \vdash C}\to_g \quad \text{se traduit en} \quad \dfrac{\dfrac{\dfrac{\begin{matrix}\vdots\\ \Gamma, A \to B;\vdash A\end{matrix} \quad \dfrac{}{\Gamma, A \to B; B \vdash B}Ax}{\Gamma, A \to B; A \to B \vdash B}\to_g}{\Gamma, A \to B;\vdash B}\mathit{Déch} \qquad \begin{matrix}\vdots\\ \Gamma, A \to B, B;\vdash C\end{matrix}}{\Gamma, A \to B;\vdash C}\mathit{Coup}_M$$

$$\dfrac{\begin{matrix}\vdots\\ \Gamma, A \vdash B\end{matrix}}{\Gamma \vdash A \to B}\to_d \quad \text{se traduit en} \quad \dfrac{\begin{matrix}\vdots\\ \Gamma, A;\vdash B\end{matrix}}{\Gamma;\vdash A \to B}\to_d$$

$$\dfrac{\begin{matrix}\vdots\\ \Gamma \vdash A\end{matrix} \quad \begin{matrix}\vdots\\ \Gamma, A \vdash B\end{matrix}}{\Gamma \vdash B}\mathit{Coupe} \quad \text{se traduit en} \quad \dfrac{\begin{matrix}\vdots\\ \Gamma;\vdash A\end{matrix} \quad \begin{matrix}\vdots\\ \Gamma, A;\vdash B\end{matrix}}{\Gamma;\vdash B}\mathit{Coup}_M$$

2.4 Le $\overline{\lambda}$-calcul

2.4.1 Les expressions du $\overline{\lambda}$-calcul

Il y a deux types de $\overline{\lambda}$-expressions qui sont mutuellement dépendantes : les $\overline{\lambda}$-termes et les listes de $\overline{\lambda}$-termes (= listes d'arguments).

Comme pour le λ-calcul, on considère un ensemble infini $\mathcal{V}$ dont les éléments sont appelés **noms de variables de terme**. Nous réservons les lettres x, y, z, ... pour noter les noms variables de terme.

L'ensemble des **$\overline{\lambda}$-expressions**, recouvrant les **$\overline{\lambda}$-termes** (ou tout simplement **termes**) et les **listes de $\overline{\lambda}$-termes** (appelées aussi **listes d'arguments**) sont définis inductivement par la grammaire suivante où x parcourt $\mathcal{V}$

$$\begin{array}{lrcl} \textbf{Termes:} & t & ::= & (x\ l) \mid (\lambda x.t) \mid (t\ l) \mid (t[x := t]) \\ \textbf{Listes d'arguments:} & l & ::= & [\,] \mid [t :: l] \mid (l\ @\ l) \mid l[x := t] \end{array}$$

Nous utilisons les lettres t, u, v,... pour noter les termes et nous utilisons la lettre l, éventuellement primée, pour noter les listes d'arguments.

L'expression "[ ]" joue le rôle de la liste vide d'arguments et l'expression "$[t :: l]$" représente la liste dont le premier argument est le terme t et le reste des arguments est dans la liste l. Quant à l'expression "$(l\ @\ l')$", elle représente l'opération explicite de concaténation des listes d'arguments l et l'.

L'expression "$(t[x := u])$" tient le rôle de l'opération explicite de substitution de x par u dans le terme t (c'est l'équivalent d'un `let x=u in t` à la ML) et l'expression "$(l[x := u])$" joue le même rôle mais pour une substitution dans la liste d'arguments l.

On abrègera communément l'expression $[t_1 :: [... :: [t_n :: [\,]]...]]$ par $[t_1; ...; t_n]$. Une expression telle que $((...(t\ t_1)\ ...)\ t_n)$ sera abrégée en $(t\ t_1\ ...\ t_n)$. Parfois $(x\ [\,])$ est simplifié en x. Aussi, les expressions $(\lambda x.t)$, $(t[x := u])$ et $(l\ @\ l')$ sont parfois écrites respectivement $\lambda x.t$, $t[x := u]$ et $l\ @\ l'$ quand il n'y a pas d'ambiguïté.

Les sous-expressions d'un terme ou d'une liste d'arguments sont définies comme à l'accoutumée sauf qu'ici, il faut tenir compte de la définition mutuelle des termes et listes d'arguments. En particulier, une liste d'arguments peut être une sous-expression d'un terme et vice-versa.

Les notions de variables liées et libres sont définies comme on le fait habituellement. Nous dirons aussi que deux $\overline{\lambda}$-expressions sont α-équivalents si elles ne diffèrent que dans les noms, supposés distincts entre eux, des variables liées. Cette notion d'équivalence n'affecte pas la structure des expressions, ce qui nous permet par la suite de considérer ces $\overline{\lambda}$-expressions à α-équivalence près.

2.4.2 Formes normales du $\overline{\lambda}$-calcul

Une $\overline{\lambda}$-expression est **normale** si elle ne contient ni opérateur de concaténation ou de substitution et si les sous-expressions de la forme $(t\ l)$ sont exclues.

Autrement dit, une $\overline{\lambda}$-expression est normale si elle est formée à partir de la grammaire suivante :

$$\begin{array}{rcl} t & ::= & (x\ l) \mid (\lambda x.t) \\ l & ::= & [\,] \mid [t :: l] \end{array}$$

Remarque : Pour ce calcul, par opposition à la syntaxe habituelle du λ-calcul, il est *directement* possible d'adopter un point de vue où les $\overline{\lambda}$-expressions normales sont les objets de base et où les autres constructeurs sont juste des notations explicites pour des fonctions sur ces objets de bases (et donc plus des constructeurs au sens strict)

Une approximation de la normalité est la normalité faible.

Un terme (resp une liste d'arguments) est qualifiée de **faiblement normal(e)** si il(elle) est de la forme $(x\ l)$ ou $\lambda x.t$ (resp [ ] ou $[t :: l]$), où t est un terme quelconque et l une liste d'arguments quelconques.

Remarque : Dans le $\overline{\lambda}$-calcul, il y a la possibilité de considérer des termes de la forme $(...(x\ [u_1])\ ...\ [u_n])$ ayant une structure similaire à celle des termes applicatifs du λ-calcul. Cependant, un tel $\overline{\lambda}$-terme n'est pas normal. De fait, sa forme normale est $(x\ [u_1; ...; u_n])$.

2.4.3 Règles de réduction du $\overline{\lambda}$-calcul

La présence d'opérateur explicite de substitution et de concaténation entraîne la présence de règles de réduction appropriées :

- β-réduction

$$\begin{array}{rcll} (\lambda x.u\ [v :: l]) & \xrightarrow{r} & (u[x := v]\ l) & \beta_{cons} \\ (\lambda x.u\ [\,]) & \xrightarrow{r} & \lambda x.u & \beta_{nil} \end{array}$$

- indication de concaténation

$$((x\ l)\ l') \xrightarrow{r} (x\ (l\ @\ l')) \quad C_{var}$$

- règles de calcul de la concaténation

$$\begin{array}{rcll} [u :: l]\ @\ l' & \xrightarrow{r} & [u :: (l\ @\ l')] & C_{cons} \\ [\,]\ @\ l' & \xrightarrow{r} & l' & C_{nil} \end{array}$$

- règles de propagation de la substitution à travers les termes faiblement normaux

$$\begin{array}{rcll} (x\ l)[x := v] & \xrightarrow{r} & (v\ l[x := v]) & S_{yes} \\ (y\ l)[x := v] & \xrightarrow{r} & (y\ l[x := v]) & S_{no} \\ (\lambda y.u)[x := v] & \xrightarrow{r} & \lambda y.(u[x := v]) & S_{\lambda} \end{array}$$

  attention à une possible capture de variable dans la règle S_λ

- règles de propagation de la substitution à travers les listes d'arguments faiblement normales

$$\begin{array}{rcll} [\,][x := v] & \xrightarrow{r} & [\,] & S_{nil} \\ [u :: l][x := v] & \xrightarrow{r} & [u[x := v] :: l[x := v]] & S_{cons} \end{array}$$

Si $u \xrightarrow{r} v$, alors u est appelé **rédex**. Nous notons $\xrightarrow{1}$ la réduction obtenue à partir de $\xrightarrow{r}$ par congruence. Puisque le système de règles de réduction est linéaire à gauche [1] et aussi sans paires critiques, en vertu du résultat de Huet dans [32], $\xrightarrow{1}$ est confluent. Par simplicité nous avons escamoté le problème de la possible capture de variable dans la règle S_λ. Une solution au problème est d'ajouter au langage un opérateur explicite de renommage des variables liées. Ceci est fait en appendice à l'occasion de la description de l'implémentation d'une procédure de normalisation pour le $\overline{\lambda}$-calcul.

Remarque : Le système de réduction que nous présentons est un système sans paire critique dont il est relativement aisé de faire une preuve de normalisation forte. Cependant, un autre système de réduction à considérer est celui que l'on obtient en ajoutant la règle $(\lambda y.u\ v)[x := w] \xrightarrow{r} ((\lambda y.u)[x := w]\ v[x := w])$. De fait, par cette extension, il devient possible de simuler la β-réduction habituelle, et donc, de voir le λ-calcul comme un sous-système strict du $\overline{\lambda}$-calcul. Cependant, la terminaison forte du système de réduction enrichi reste à l'état de conjecture.

2.5 Le calcul LJT

Un **environnement** est un ensemble de formules nommées. Plusieurs occurrences de la même formule peuvent apparaître dans un environnement mais ce doit être à chaque fois avec un nom différent.

Les lettres grecques majuscules Γ, Δ, Ω,... sont réservées pour les environnements. Si Γ et Δ ont une intersection vide, la notation Γ, Δ a du sens et représente l'union ensembliste de Γ et Δ.

Dans la pratique, sauf pour souligner une ambiguïté, nous n'écrivons pas les noms des formules nommées. Aussi, une notation comme Δ, A désigne l'union de Δ et de $\{A\}$ où il est implicite que le nom de A n'interfère pas avec une formule A de même nom dans Δ.

[1] Précisons qu'il nous faut considérer les règles S_{yes}, S_{no} et S_λ comme des schémas de règles en x et y dont on considère les instances lorsques x et y parcourt $\mathcal{V}$. Ainsi, on a effectivement linéarité à gauche

2.5.1 Elimination des coupures

Nous disons que deux preuves sont **équivalentes** si elles ne diffèrent que par le nom des formules nommées impliquées dans la preuve ou que par l'ajout de formules non utilisées dans la partie environnement des séquents (pas utilisées voulant dire qu'on peut suivre leur trace dans la partie environnement des séquents jusqu'aux axiomes).

Dans la suite, nous considérons les preuves à cette équivalence près-là. En particulier, si p est une preuve de $\Gamma; \Pi \vdash A$, alors, pour toute formule nommée B n'interférant pas avec une même formule nommée déjà présente dans p, cette preuve p est aussi considérée comme une preuve de $\Gamma, B; \Pi \vdash A$.

Nous qualifions de **normale** une preuve sans coupure.

Proposition 1 *(Elimination forte et confluente des coupures)*
Il existe une PEC confluente et fortement normalisable pour LJT.

Un système de réécriture définissant une telle PEC est décrit ci-dessous. Il est facile de constater son exhaustivité, puisque toutes les configurations possibles avec une coupure sont prises en compte.

La confluence, comme pour le système de réduction du $\overline{\lambda}$-calcul provient de la linéarité à gauche et de l'absence de paires critiques du système de réécriture. Quant à la normalisation forte elle est montrée dans une prochaine section.

Réduction de la règle $Coup_T$

- contrepartie logique de la β-réduction

$$\dfrac{\dfrac{\Gamma, A; \vdash B}{\Gamma; \vdash A \to B}\to_d \quad \dfrac{\Gamma; \vdash A \quad \Gamma; B \vdash C}{\Gamma; A \to B \vdash C}\to_g}{\Gamma; \vdash C}\, Coup_T \quad \overset{\text{LJT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma; \vdash A \quad \Gamma, A; \vdash B}{\Gamma; \vdash B}\, Coup_M \quad \Gamma; B \vdash C}{\Gamma; \vdash C}\, Coup_T$$

$$\dfrac{\dfrac{\Gamma, A; \vdash B}{\Gamma; \vdash A \to B}\to_d \quad \dfrac{}{\Gamma; A \to B \vdash A \to B}\, Ax}{\Gamma; \vdash A \to B}\, Coup_T \quad \overset{\text{LJT}}{\longrightarrow} \quad \dfrac{\Gamma, A; \vdash B}{\Gamma; \vdash A \to B}\to_d$$

- contrepartie logique de l'indication de concaténation

$$\dfrac{\dfrac{\Gamma, B; B \vdash A}{\Gamma, B; \vdash A}\, D\acute{e}ch \quad \Gamma, B; A \vdash C}{\Gamma, B; \vdash C}\, Coup_T \quad \overset{\text{LJT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma, B; B \vdash A \quad \Gamma, B; A \vdash C}{\Gamma, B; B \vdash C}\, Coup_T}{\Gamma, B; \vdash C}\, D\acute{e}ch$$

- contrepartie logique du calcul de la concaténation

$$\dfrac{\dfrac{\Gamma; \vdash D \quad \Gamma; B \vdash A}{\Gamma; D \to B \vdash A}\to_g \quad \Gamma; A \vdash C}{\Gamma; D \to B \vdash C}\, Coup_T \quad \overset{\text{LJT}}{\longrightarrow} \quad \dfrac{\Gamma; \vdash D \quad \dfrac{\Gamma; B \vdash A \quad \Gamma; A \vdash C}{\Gamma; B \vdash C}\, Coup_T}{\Gamma; D \to B \vdash C}\to_g$$

$$\dfrac{\dfrac{}{\Gamma; A \vdash A}\, Ax \quad \Gamma; A \vdash C}{\Gamma; A \vdash C}\, Coup_T \quad \overset{\text{LJT}}{\longrightarrow} \quad \Gamma; A \vdash C$$

Réduction de la règle $Coup_M$

- contrepartie logique de la propagation de la substitution à travers les termes faiblement normaux

$$\frac{\Gamma;\vdash A \quad \dfrac{\Gamma,A;A\vdash C}{\Gamma,A;\vdash C}\,\textit{Déch}}{\Gamma;\vdash C}\,Coup_M \quad \overset{\mathrm{LJT}}{\longrightarrow} \quad \frac{\Gamma;\vdash A \quad \dfrac{\Gamma;\vdash A \quad \Gamma,A;A\vdash C}{\Gamma;A\vdash C}\,Coup_M}{\Gamma;\vdash C}\,Coup_T$$

$$\frac{\Gamma,B;\vdash A \quad \dfrac{\Gamma,A,B;B\vdash C}{\Gamma,A,B;\vdash C}\,\textit{Déch}}{\Gamma,B;\vdash C}\,Coup_M \quad \overset{\mathrm{LJT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma,B;\vdash A \quad \Gamma,A,B;B\vdash C}{\Gamma,B;B\vdash C}\,Coup_M}{\Gamma,B;\vdash C}\,\textit{Déch}$$

$$\frac{\Gamma;\vdash A \quad \dfrac{\Gamma,A,B;\vdash C}{\Gamma,A;\vdash B\to C}\,\to_d}{\Gamma;\vdash B\to C}\,Coup_M \quad \overset{\mathrm{LJT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma,B;\vdash A \quad \Gamma,A,B;\vdash C}{\Gamma,B;\vdash C}\,Coup_M}{\Gamma;\vdash B\to C}\,\to_d$$

Pour cette règle, on considère la preuve de $\Gamma;\vdash A$ comme une preuve de $\Gamma,B;\vdash A$. S'il advenait que B soit déjà présent avec le même nom quelque part dans cette preuve, il faudrait y changer le nom de cette formule partout où elle intervient.

- contrepartie logique de la propagation de la substitution à travers les listes d'arguments faiblement normales

$$\frac{\Gamma;\vdash A \quad \dfrac{\Gamma,A;\vdash B \quad \Gamma,A;C\vdash D}{\Gamma,A;B\to C\vdash D}\,\to_g}{\Gamma;B\to C\vdash D}\,Coup_M \quad \overset{\mathrm{LJT}}{\longrightarrow} \quad \frac{\dfrac{\Gamma;\vdash A \quad \Gamma,A;\vdash B}{\Gamma;\vdash B}\,Coup_M \quad \dfrac{\Gamma;\vdash A \quad \Gamma,A;C\vdash D}{\Gamma;C\vdash D}\,Coup_M}{\Gamma;B\to C\vdash D}\,\to_g$$

$$\frac{\Gamma;\vdash A \quad \dfrac{}{\Gamma,A;B\vdash B}\,Ax}{\Gamma;B\vdash B}\,Coup_M \quad \overset{\mathrm{LJT}}{\longrightarrow} \quad \frac{}{\Gamma;B\vdash B}\,Ax$$

2.6 L'assignement des preuves de LJT par des $\overline{\lambda}$-expressions

Les preuves de LJT ont une structure de $\overline{\lambda}$-expression. Nous mettons ce fait en valeur par un assignement de $\overline{\lambda}$-expressions aux preuves de LJT. Les règles de réduction du $\overline{\lambda}$-calcul et les règles de réécriture pour l'élimination des coupures s'identifieront par cette assignement.

Une **déclaration**, écrite sous la forme $x\!:\!A$ est la paire d'une variable et d'une formule.

Nous gardons le vocable d'**environnement** pour désigner un ensemble de déclarations dont aucunes formules ne partagent le même nom de variable. Nous gardons aussi les lettres grecques Γ, Δ, Ω pour désigner ces environnements.

Jusqu'à la fin de cette section, nous adoptons une notation alternative pour les listes d'arguments qui insiste sur leur caractère de terme potentiel.

Un **contexte applicatif** est une liste d'arguments écrite sous la forme $(.\ \ l)$ où $.$ est un symbole spécialement introduit pour l'occasion. On appelle aussi **déclaration de place** une formule écrite sous la forme $. : A$.

Nous exprimons l'assignement à l'aide de jugements.

Un **jugement** est une expression syntaxique de la forme $\Gamma; \Pi \vdash t : A$. Dans cette écriture, Γ est un environnement, Π est un ensemble soit vide soit contenant une formule qui est notée sous la forme d'une déclaration de place, t est une $\overline{\lambda}$-expression et A est une formule.

Autrement dit les séquents avec bénitier vide sont interprétés par des $\overline{\lambda}$-termes et les séquents avec bénitier non vide sont interprétés par des listes d'arguments, notées sous la forme de contexte applicatif.

Règles de formation de contextes applicatifs

Liste vide d'arguments
$$\frac{}{\Gamma; . : A \vdash (.\ \ [\,]) : A}\ Ax$$

Ajout d'un argument à une liste
$$\frac{\Gamma; \vdash u : A \quad \Gamma; . : B \vdash (.\ \ l) : C}{\Gamma; . : A \to B \vdash (.\ \ [u :: l]) : C}\ \to_g$$

Concaténation de listes d'arguments
$$\frac{\Gamma; . : C \vdash (.\ \ l) : A \quad \Gamma; . : A \vdash (.\ \ l') : B}{\Gamma; . : C \vdash (.\ \ (l\ @\ l')) : B}\ Coup_T$$

Substitution dans une liste d'arguments
$$\frac{\Gamma; \vdash u : A \quad \Gamma, x : A; . : C \vdash (.\ \ l) : B}{\Gamma; . : C \vdash (.\ \ l[x := u]) : B}\ Coup_M$$

Règles de formation de termes

Terme applicatif faiblement normal
$$\frac{\Gamma, x : A; . : A \vdash (.\ \ l) : B}{\Gamma, x : A; \vdash (x\ l) : B}\ Déch$$

Abstraction
$$\frac{\Gamma, x : A; \vdash u : B}{\Gamma; \vdash \lambda x.u : A \to B}\ \to_d$$

Terme applicatif non faiblement normal
$$\frac{\Gamma; \vdash u : A \quad \Gamma; . : A \vdash (.\ \ l) : B}{\Gamma; \vdash (u\ l) : B}\ Coup_T$$

Substitution dans un terme
$$\frac{\Gamma; \vdash u : A \quad \Gamma, x : A; \vdash v : B}{\Gamma; \vdash v[x := u] : B}\ Coup_M$$

Remarque : Les règles avec un bénitier non vide sont polymorphiques par rapport à la formule qui se trouve dans le bénitier. Ainsi, il y a une relation forte entre un jugement

$$\Gamma; . : A_1 \to ... \to A_n \to B \vdash (.\ \ [u_1; ...; u_n]) : B$$

et un jugement

$$\Gamma \vdash [u_1; ...; u_n] : A_1 \wedge ... \wedge A_n$$

où $A_1 \wedge ... \wedge A_n$ est défini comme étant $\forall B.(A_1 \to ... \to A_n \to B) \to B$ (codage des n-uplets au second ordre).

Une $\overline{\lambda}$-expression e telle qu'on aie $x_1 : A_1, ..., x_n : A_n \vdash e : A$ pour des noms de variables x_1, ... x_n et pour des formules A_1, ..., A_n, A bien choisies est dit **simplement typé de type** A, ou plus brièvement, **typable par** A.

2.7 Termination forte de la réduction

De par l'isomorphisme, la termination forte de l'élimination des coupures pour LJT et la termination forte de la réduction pour les $\overline{\lambda}$-expressions typables sont équivalentes. Nous le montrons pour les $\overline{\lambda}$-expressions typables, ce qui permet une présentation plus légère que si on avait choisi les notations du calcul des séquents.

La preuve de termination forte

Soit e une $\overline{\lambda}$-expression et $\xrightarrow{R}$ une notion de réduction. On dit que e est fortement normalisable selon $\xrightarrow{R}$ dans les cas suivants :

- e n'est pas reductible par $\xrightarrow{R}$
- pour tout e' tel que $e \xrightarrow{R} e'$, on a e' fortement normalisable

Remarque : Les preuves de forte normalisabilité ont une structure inductive. On peut donc raisonner sur elles par induction structurelle.

Nous montrons donc qu'une $\overline{\lambda}$-expression typable est fortement normalisable selon la réduction $\xrightarrow{1}$ et pour ceci, nous montrons quelque chose de plus fort, la forte E-normalisabilité que les opérations de construction des expressions préservent.

On définit une réduction $\xrightarrow{h}$ qui supprime le constructeur de tête d'une $\overline{\lambda}$-expression faiblement normale. Cette réduction préserve la typabilité mais pas le typage.

La réduction $\xrightarrow{h}$ est satisfaite dans les seuls cas suivants :

$$\begin{array}{rcl} \lambda x.u & \xrightarrow{h} & u \\ (x\ l) & \xrightarrow{h} & l \\ [u :: l] & \xrightarrow{h} & u \\ [u :: l] & \xrightarrow{h} & l \end{array}$$

où u parcourt l'ensemble des $\overline{\lambda}$-termes et l parcourt l'ensemble des listes d'arguments.

On note $\xrightarrow{E}$, et on appelle E-réduction, la réduction définie par $e \xrightarrow{E} e'$ soit parce que $e \xrightarrow{1} e'$, soit parce que $e \xrightarrow{h} e'$ (sans considérer l'extension de cette réduction par congruence). On dit que e est **fortement E-normalisable** (en abrégé FEN), ou encore qu'il satisfait **la forte E-normalisabilité** (en abrégé FEN aussi) s'il est fortement normalisable selon $\xrightarrow{E}$.

Lemme 1 *Si le $\overline{\lambda}$-terme u et la liste d'argument l satisfont la FEN alors $\lambda x.u$, $(x\ l)$ et $[u :: l]$ aussi.*

Preuve : Par induction sur les preuves de FEN pour u et l. Regardons le cas de $[u :: l]$. Si $[u :: l] \xrightarrow{E} e'$ alors, ou bien $[u :: l] \xrightarrow{h} e'$, c'est-à-dire que e' peut être soit u soit l, et en ce cas, par hypothèse, e' satisfait la FEN, ou bien $[u :: l] \xrightarrow{1} e'$, ce qui veut dire que e' est soit $[u' :: l]$ avec $u \xrightarrow{1} u'$, soit $[u :: l']$ avec $l \xrightarrow{1} l'$, et en ce cas e' satisfait la FEN par hypothèse d'induction. Dans tous les cas possibles de réduction de e, celui-ci se réduit en une expression FEN. e est donc lui-même FEN. ∎

Lemme 2 *Soit e et u des $\overline{\lambda}$-expressions FEN. Si, pour tout l FEN, la typabilité de $(u\ l)$ entraine sa FEN, alors la typabilité de $e[x := u]$ entraine à son tour sa FEN.*

Preuve : On raisonne par induction sur la preuve de FEN pour e puis par induction sur la preuve de FEN pour u.

Supposons que $e[x := u] \xrightarrow{1} w$. Si la réduction affecte un rédex dans u alors w a la forme $e[x := u']$ avec $u \xrightarrow{1} u'$. La preuve de FEN pour u' est plus petite que celle pour u d'où, par hypothèse d'induction, $e[x := u']$ est FEN. Pareillement si la réduction est dans e.

Il reste le cas où $e[x := u]$ est lui-même un rédex et où c'est ce rédex qui est réduit. On envisage alors les différentes formes possibles pour e qui peut être soit un terme soit une liste d'arguments.

- Le cas où e est $(x\ l')$ auquel cas w désigne $(u\ l'[x := u])$, est le plus délicat. Mais puisque que $e \xrightarrow{h} l'$, l' a une plus petite preuve de FEN que e. D'où, par hypothèse d'induction, $l'[x := u]$ est FEN. Et comme on a supposé que pour tout l FEN, $(u\ l)$ était FEN, on en déduit que $(u\ l'[x := u])$ est FEN.
- Si e est $(y\ l)$ alors w est $(y\ l[x := u])$. Là encore, $l[x := u]$ est FEN par hypothèse d'induction et, par le lemme 1, w est FEN.
- Si e est le terme $\lambda y.v$, à un renommage près de y dans $\lambda y.v$, ce qui n'affecte pas la preuve de FEN, nous pouvons supposer que y et x sont des noms de variables distinctes. On peut donc affirmer que w est $\lambda y.(v[x := u])$. Puisque $\lambda y.v \xrightarrow{E} v$, par hypothèse d'induction, $(v[x := u])$ est FEN et par le lemme 1, w est FEN.
- Si e est $[v :: l]$ alors w désigne $[v[x := u] :: l[x := u]]$. Mais on a à la fois $[v :: l] \xrightarrow{E} v$ et $[v :: l] \xrightarrow{E} l$, d'où, par hypothèse d'induction, à la fois $v[x := u]$ et $l[x := u]$ sont FEN et par le lemme 1, w est aussi FEN.
- Si e est $[\,]$ alors w est $[\,]$ auquel cas il est bien sûr FEN.

Ainsi, quelque soit la forme de e, les réduits de $e[x := u]$ sont FEN. Cela suffit pour dire que $e[x := u]$ est lui-même FEN.

■

Lemme 3 *Soit A une formule. Soit e une $\overline{\lambda}$-expression FEN et typable par A et soit l_2 une liste d'arguments FEN. Si l'expression $(e\ l_2)$ (cas où e est un terme) ou bien $e\ @\ l_2$ (cas où e est une liste d'arguments) est typable, alors elle est FEN.*

PREUVE : On raisonne par induction sur A, puis sur la preuve de FEN de e, puis sur la preuve de FEN de l_2.

Supposons que $(e\ l_2) \xrightarrow{1} w$ (cas où e est un terme) ou bien $e\ @\ l_2 \xrightarrow{1} w$ (cas où e est une liste d'arguments).

Si la réduction affecte un rédex dans e alors w a la forme $(e'\ l_2)$ ou bien $e'\ @\ l_2$ avec $e \xrightarrow{1} e'$. Puisque la preuve de FEN pour e' est plus petite que celle de FEN pour e, par hypothèse d'induction, w est FEN. Pareillement si la réduction est dans l_2.

Mais $(e\ l_2)$ ou bien $e\ @\ l_2$ peut être lui-même un rédex qui est celui réduit. On envisage alors les différentes formes possibles pour e.

- Le cas le plus subtil est celui où e a la forme $\lambda x.u$ et où l_2 représente $[v :: l]$. Dans un tel cas, w désigne $(u[x := v]\ l)$ et, de plus, A doit avoir la forme $B \to C$, tandis que v est typable par B. Puisque B est plus petit que A, par hypothèse d'induction, la typabilité de $(v\ l')$ entraine sa FEN quelque soit l' FEN. On peut donc appliquer le lemme 2 pour inférer que $u[x := v]$ est FEN. Mais ce dernier est typable par C qui est aussi plus petit que A. Par hypothèse d'induction encore, $(u[x := v]\ l)$ est donc FEN.
- Si e est $(x\ l)$ alors w désigne $(x\ (l\ @\ l_2))$. Mais $(x\ l) \xrightarrow{E} l$ d'où, par hypothèse d'induction, $(l\ @\ l_2)$ est FEN. Par le lemme 1, w est FEN.
- Si e est $\lambda x.u$ et l_2 désigne $[\,]$ alors w représente e qui, par hypothèse, est FEN.
- Si e est $[\,]$ alors w désigne l_2 qui est trivialement FEN.
- Si e est $[v :: l]$ alors w désigne $[v :: (l\ @\ l_2)]$. Mais $[v :: l] \xrightarrow{E} l$ d'où, par hypothèse d'induction, $l\ @\ l_2$ est FEN, et, par le lemme 1, w est FEN.

Ainsi, toutes les manières de réduire $(e\ l_2)$ ou bien $e\ @\ l_2$ conduisent à une expression FEN. L'expression $(e\ l_2)$ (cas où e est un terme) ou bien $e\ @\ l_2$ (cas où e est une liste d'arguments) est donc elle-même FEN.

■

Proposition 2 *Les $\overline{\lambda}$-expressions typables sont FEN.*

Preuve : Soit e une $\overline{\lambda}$-expression typable. On raisonne par induction sur e. Les cas λv, $(p\ l')$ et $(v :: l')$ s'obtiennent directement par le lemme 1. Les cas $(u\ l)$ et $(l_1\ @\ l_2)$ s'obtiennent par le lemme 3. Quant aux cas $v[x := u]$ et $l[x := u]$, ils s'obtiennent par le lemme 2 appliqué au lemme 3. ■

La forte E-normalisabilité entraine trivialement la forte normalisabilité.

Corollaire 1 *Les $\overline{\lambda}$-expressions simplement typées sont fortement normalisables.*

Remarques : 1) Pour obtenir cette preuve, nous avons généralisé une preuve de terminaison de l'élimination des coupures par la stratégie de réduction de la coupure la plus externe d'abord, mise au point par Thierry Coquand (et reproduite dans le chapitre 6). Cette généralisation nous fournit une preuve fonctionnant par induction sur la forme normale des termes puis sur leur preuve de forte normalisation. L'idée de mêler ces deux inductions en une induction sur la preuve de forte E-normalisation est de nouveau due à Thierry Coquand. La preuve obtenue s'avère alors similaire à la preuve d'élimination forte des coupures de Dragalin [15], à ceci près que dans cette dernière, on raisonne non pas structurellement mais sur la longueur de la preuve de forte E-normalisation.

2) Il serait intéressant aussi de prouver la terminaison forte d'un système de réduction intégrant la règle supplémentaire $(\lambda y.u\ v)[x := w] \xrightarrow{r} ((\lambda y.u)[x := w]\ v[x := w])$, car, à ce moment-là, on obtiendrait la terminaison forte de la β-réduction dans le λ-calcul comme conséquence. Si l'on se base sur le résultat de terminaison forte pour le système de réduction du λv-calcul [2], il semble raisonnable de conjecturer la terminaison forte du système enrichi. Cependant, sera-t-il possible de trouver une preuve qui ne présuppose pas la normalisation de la β-réduction (comme c'est le cas pour la preuve de terminaison forte du λv-calcul donnée dans [2]).

2.8 Travaux connexes

Les premières relations entre LJ et le λ-calcul apparaîssent implicitement dans Gentzen [20] via l'équivalence logique entre LJ et la déduction naturelle. La correspondance entre LJ et la déduction naturelle a ensuite été rendue plus précise par Prawitz dans [50] qui donna un procédé simple de transformation des preuves de l'une vers l'autre syntaxe et réciproquement.

Zucker dans [60] déduisit un théorème d'élimination forte des coupures dans LJ à partir de cette correspondance. Cependant, il raisonnait modulo des permutations des coupures (les étapes de commutation du chapitre 1), si bien que l'on peut dire que son résultat signifie la forte normalisabilité des étapes principales de réduction (telles que définies dans le chapitre 1, autrement dit de la β-réduction).

Le résultat de Zucker ne valait que pour le fragment de LJ avec implication, conjonction et quantification universelle. L'extension à la disjonction et la quantification existentielle a été faite par Pottinger dans [49].

D'autres interprétations calculatoires des calculs des séquents LJ et **LK** apparaîssent dans Gallier [19] sur des idées de Breazu Tanen, dans Wadler [59] et dans Breazu Tanen *et al* [7]. Pour ces trois travaux, les règles d'introduction gauche sont interprétées par des constructeurs de motifs.

Enfin, une preuve similaire d'élimination forte et non déterministe des coupures dans le calcul des séquents **LK**, en suivant les règles de réduction originelle de Gentzen, a été faite par Dragalin [15] et adaptée à la version calcul des séquents de la logique linéaire par Roorda [58].

Chapitre 3

LKT et le $\overline{\lambda}\mu$-calcul

Un isomorphisme semblable à celui entre le $\overline{\lambda}$-calcul et le calcul LJT pour la logique minimale est possible pour un calcul des séquents classique. Pour cela on peut considérer l'opérateur μ de Parigot, ou alors l'opérateur C de Felleisen.

Le $\lambda\mu$-calcul de Parigot [48] étend l'isomorphisme de Curry-Howard-de Bruijn à une version classique de la déduction naturelle où l'aspect classique est obtenu en autorisant plusieurs conclusions. Autant les formes normales du λ-calcul simplement typé sont en bijection avec les preuves sans coupures de LJT, autant les formes normales du $\lambda\mu$-calcul simplement typé sont en bijection avec les preuves sans coupures de LKT. Ce calcul LKT est l'extension classique de LJT où, là aussi, le côté classique est obtenu en autorisant plusieurs conclusions. La définition de LKT est due à Joinet et à Schellinx dans leur thèse de doctorat [35, 51] et apparaît aussi dans Danos *et al* [14] où le lien entre LKT et le $\lambda\mu$-calculus est d'ailleurs évoqué [1].

L'opérateur C de Felleisen, défini dans [16], est un opérateur qui, doté de règles adéquates, permet d'enrichir le λ-calcul avec quelques caractéristiques impératives telles que sauts, traitements d'exceptions, calculs d'indices de séquentialité, ... Il a été observé par Griffin [29] et clarifié par Murthy [45] que, dans un cadre typé, cet opérateur reçoit un type de la forme $\neg\neg A \rightarrow A$, permettant d'étendre l'isomorphisme de Curry-Howard-de Bruijn à la déduction naturelle classique où l'aspect classique est, cette fois, obtenu en ajoutant l'axiome $\neg\neg A \rightarrow A$ pour tout A, autrement dit axiome d'élimination de la double négation.

Dans ce chapitre, nous considérons donc deux extensions du $\overline{\lambda}$-calcul. La première considère l'opérateur μ du $\lambda\mu$-calcul et fournit un isomorphisme avec le calcul LKT. La seconde considère l'opérateur C de Felleisen et fournit un isomorphisme avec LJT + $\neg\neg A \rightarrow A$. Les règles de réduction obtenues avec cette deuxième extension sont néanmoins un peu plus lourdes que celles impliquées par l'extension avec l'opérateur μ.

3.1 Le $\overline{\lambda}\mu$-calcul

En plus de l'ensemble $\mathcal{V}$ des variables de termes, on admet l'existence d'un ensemble infini $\mathcal{V}_C$ dont les éléments sont appelés **noms de μ-variables**. On utilise les lettres grecques minuscules α, β, γ, ... pour désigner ces noms de μ-variables.

L'ensemble des **$\overline{\lambda}\mu$-expressions**, recouvrant les **$\overline{\lambda}\mu$-termes** (ou **termes**) et les **listes de $\overline{\lambda}\mu$-termes** (ou aussi **listes d'arguments**) est défini inductivement par la grammaire suivante où x parcourt $\mathcal{V}$ et α, β parcourent $\mathcal{V}_C$:

$$\begin{array}{llll} \textbf{Termes:} & t & ::= & (x\ l) \mid (\lambda x.t) \mid (t\ l) \mid (t[x := t]) \mid \mu\beta[\alpha]t \mid t[[\alpha]\chi := [\alpha](\chi\ l)] \\ \textbf{Listes d'arguments:} & l & ::= & [\,] \mid [t :: l] \mid (l\ @\ l) \mid l[x := t] \mid l[[\alpha]\chi := [\alpha](\chi\ l)] \end{array}$$

On réutilise pour désigner les $\overline{\lambda}\mu$-termes les mêmes lettres t, u, v,... que pour les $\overline{\lambda}$-termes et, pareillement, les notations l, l', ... pour désigner les listes d'arguments.

Schématiquement, un $\overline{\lambda}\mu$-terme gère plusieurs $\overline{\lambda}$-termes imbriqués. Chacun des termes imbriqués est repéré par un nom de conclusion α, β, etc. Plusieurs $\overline{\lambda}$-termes peuvent partager le même nom. A chaque instant, il y a une famille de ces $\overline{\lambda}$-termes partageant le même nom qui est active et le rôle de l'expression $\mu\beta[\alpha]t$ est de désactiver le terme courant t en le cachant sous le nom α et d'activer la famille de termes ayant le nom β. Quant à l'expression $t[[\alpha]\chi := [\alpha](\chi\ l)]$, elle désigne l'opération explicite de diffusion des arguments l vers chacun des sous-termes de nom α (χ est juste un symbole formel sensé représenter le terme se trouvant dans le champ de $[\alpha]$). Similairement pour $l[[\alpha]\chi := [\alpha](\chi\ l)]$ à l'intérieur d'une liste d'arguments.

Dans l'optique où les expressions normales sont les objets de base et où les autres constructeurs ne sont que des symboles explicites de fonctions sur ces objets de bases, seul $\mu\beta[\alpha]t$ est à part entière un nouveau constructeur d'expressions.

Une $\overline{\lambda}\mu$-expression est normale si elle fait partie des expressions engendrées par cette grammaire :

$$\begin{array}{lll} t & ::= & (x\ l) \mid (\lambda x.t) \mid \mu\beta[\alpha]t \\ l & ::= & [\,] \mid [t :: l] \end{array}$$

[1] De la même manière que LJT est une restriction de LJ qui se comporte bien relativement au plongement dans la logique linéaire, le calcul LKT est une restriction de LK dont les preuves ont une structure de preuves de la logique linéaire.

Si un nom β apparaît dans le champ d'un $\mu\beta[\gamma]$, on dit que la μ-variable de nom β est liée, au même titre qu'une variable de terme de nom x est liée dans le champ d'un λx. On dit que deux $\overline{\lambda}\mu$-expressions sont α-équivalentes si elles ne diffèrent que dans les noms, supposés distincts entre eux, des variables liées. Cette notion d'équivalence n'affecte pas la structure du terme et, dans la suite, on considère les $\overline{\lambda}\mu$-expressions à α-équivalence près.

Règles de réduction

En plus des règles de réduction du $\overline{\lambda}$-calcul, il y a, dans le $\overline{\lambda}\mu$-calcul, les règles de réduction suivantes :

- μ-réduction

$$(\mu\beta[\alpha]u\ l) \quad \xrightarrow{r} \quad \mu\beta[\alpha](u[[\beta]\chi := [\beta](\chi\ l)]) \quad \mu$$

- règles de diffusion d'une liste d'arguments à travers un terme

$$\begin{array}{rcll}
(\mu\beta[\alpha]u)[[\alpha]\chi := [\alpha](\chi\ l)] & \xrightarrow{r} & \mu\beta[\alpha](u[[\alpha]\chi := [\alpha](\chi\ l)]\ l) & D_{yes} \\
(\mu\beta[\gamma]u)[[\alpha]\chi := [\alpha](\chi\ l)] & \xrightarrow{r} & \mu\beta[\gamma](u[[\alpha]\chi := [\alpha](\chi\ l)]) & D_{no} \\
(\lambda y.u)[[\alpha]\chi := [\alpha](\chi\ l)] & \xrightarrow{r} & \lambda y.(u[[\alpha]\chi := [\alpha](\chi\ l)]) & D_{\lambda} \\
(y\ l')[[\alpha]\chi := [\alpha](\chi\ l)] & \xrightarrow{r} & (y\ (l'[[\alpha]\chi := [\alpha](\chi\ l)])) & D_{app}
\end{array}$$

- règles de diffusion d'une liste d'arguments à travers une liste d'arguments

$$\begin{array}{rcll}
([u :: l'])[[\alpha]\chi := [\alpha](\chi\ l)] & \xrightarrow{r} & [u[[\alpha]\chi := [\alpha](\chi\ l)] :: l'[[\alpha]\chi := [\alpha](\chi\ l)]] & D_{cons} \\
([\,])[[\alpha]\chi := [\alpha](\chi\ l)] & \xrightarrow{r} & [\,] & D_{nil}
\end{array}$$

  attention à une possible capture de la μ-variable de nom β à l'intérieur de l pour les règles D_{yes} and D_{no} et de la variable de nom y dans la règle D_{λ}

- règle de propagation de la substitution à travers les termes fraîchement nommés

$$(\mu\beta[\alpha]u)[x := v] \quad \xrightarrow{r} \quad \mu\beta[\alpha](u[x := v]) \quad S_{\mu}$$

3.2 Le calcul LKT

L'ensemble des **formules** est maintenant défini par la grammaire suivante :

$$A \quad ::= \quad X \mid A \to A \mid \bot$$

Un **séquent classique** a la forme $\Gamma;\Pi \vdash \Delta;A$. La place à la droite du " ;" de gauche est le bénitier. Celui-ci peut être vide ou bien contenir une unique formule. Π est une telle notation pour dire "rien ou bien une seule formule". La place à la droite du " ;" de droite est la place active. Celle-ci contient une unique formule appelée **conclusion courante**. Γ est un environnement de formules nommées par des noms de $\mathcal{V}$ et Δ est un environnement de formules nommées par des noms de $\mathcal{V}_C$.

La place où se trouve A est la place intuitionniste. Ainsi, LJT peut être vu comme le calcul LKT où l'on impose à Δ d'être toujours vide. On peut d'ailleurs constater que les règles de LKT n'affectant pas Δ sont aussi des règles de LJT.

règle d'axiome

$$\frac{}{\Gamma; A \vdash \Delta; A}\ Ax$$

règle de déchargement

$$\frac{\Gamma, A; A \vdash \Delta; B}{\Gamma, A; \vdash \Delta; B}\ D\acute{e}ch$$

règle d'introduction gauche de $\rightarrow$

$$\frac{\Gamma; \vdash \Delta; A \quad \Gamma; B \vdash \Delta; C}{\Gamma; A \rightarrow B \vdash \Delta; C} \rightarrow_g$$

règle d'introduction droite de $\rightarrow$

$$\frac{\Gamma, A; \vdash \Delta; B}{\Gamma; \vdash \Delta; A \rightarrow B} \rightarrow_d$$

règle de coupure de tête

$$\frac{\Gamma; \Pi \vdash \Delta; A \quad \Gamma; A \vdash \Delta; B}{\Gamma; \Pi \vdash \Delta; B}\ Coup_T$$

règle de coupure médiane

$$\frac{\Gamma; \vdash \Delta; A \quad \Gamma, A; \Pi \vdash \Delta; B}{\Gamma; \Pi \vdash \Delta; A}\ Coup_M$$

règle de changement de conclusion

$$\frac{\Gamma; \vdash \Delta, A, B; A}{\Gamma; \vdash \Delta, A; B}\ Chg$$

règle de μ-coupure

$$\frac{\Gamma; \Pi \vdash \Delta, A; C \quad \Gamma; A \vdash \Delta; B}{\Gamma; \Pi \vdash \Delta, B; C}\ Coup_\mu$$

Nous disons que deux preuves sont **équivalentes** si elles ne diffèrent que par le nom des formules nommées impliquées dans la preuve ou que par l'ajout ou le retrait dans les environnements de formules nommées qui n'ont pas été déchargées (i.e. qui n'ont pas été nommées au moment d'une règle de déchargement ou de changement de conclusion).

Dans la suite, nous considérons les preuves à cette équivalence près-là. En particulier, si p est une preuve de $\Gamma; \Pi \vdash \Delta; A$, alors, pour toute formule nommée B n'interférant pas avec une même formule nommée déjà présente dans p, cette preuve p est aussi considérée comme une preuve de $\Gamma, B; \Pi \vdash \Delta; A$ et comme une preuve de $\Gamma; \Pi \vdash \Delta, B; A$.

Elimination des coupures

Nous qualifions de **normale** une preuve sans coupure.

Proposition 3 *(Elimination forte et confluente des coupures)*
Il existe une PEC confluente et fortement normalisable pour LJT.

Un système exhaustif de réécriture définissant une telle PEC est définissable à partir des règles de réécriture déjà donnée pour LJT. Il suffit de généraliser celles-ci par la présence d'un environnement de conclusion Δ et de considérer les règles supplémentaires données ci-dessous.

La confluence, comme pour le système de réduction du $\overline{\lambda}$-calcul provient de la linéarité à gauche et de l'absence de paires critiques du système de réécriture. Quant à la normalisation forte, elle est montrée dans une prochaine section.

- contrepartie logique de la μ-réduction

$$\frac{\dfrac{\Gamma; \vdash \Delta, A, B; A}{\Gamma; \vdash \Delta, A; B}\ Chg \qquad \Gamma; B \vdash \Delta, A; C}{z \quad \Gamma; \vdash \Delta, A; C}\ Coup_T \quad \overset{\text{LKT}}{\longrightarrow} \quad \frac{\dfrac{\Gamma; \vdash \Delta, A, B; A \quad \Gamma; B \vdash \Delta, A; C}{\Gamma; \vdash \Delta, A, C; A}\ Coup_\mu}{\Gamma; \vdash \Delta, A; C}\ Chg$$

- contrepartie logique de la diffusion d'une liste d'arguments à travers un terme

$$\dfrac{\dfrac{\Gamma;\vdash\Delta,A,B;A}{\Gamma;\vdash\Delta,A;B}\,Chg\quad \Gamma;A\vdash\Delta;C}{\Gamma;\vdash\Delta,C;B}\,Coup_\mu \quad\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\dfrac{\Gamma;\vdash\Delta,A,B;A\quad\Gamma;A\vdash\Delta,B;C}{\Gamma;\vdash\Delta,C,B;A}\,Coup_\mu\quad \Gamma;A\vdash\Delta,C,B;C}{\Gamma;\vdash\Delta,C,B;C}\,Coup_T}{\Gamma;\vdash\Delta,C;B}\,Chg$$

Pour cette règle, on considère la preuve de $\Gamma;A\vdash\Delta;C$ et comme une preuve de $\Gamma;A\vdash\Delta,B;C$ et comme une preuve de $\Gamma;A\vdash\Delta,C,B;C$. S'il advenait que B (et C) soit déjà présent avec le même nom quelque part dans ces preuves, il faudrait y changer le nom de cette formule partout où elle intervient.

$$\dfrac{\dfrac{\Gamma;\vdash\Delta,D,A,B;D}{\Gamma;\vdash\Delta,D,A;B}\,Chg\quad \Gamma;A\vdash\Delta,D;C}{\Gamma;\vdash\Delta,D,C;B}\,Coup_\mu \quad\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\Gamma;\vdash\Delta,D,A,B;D\quad\Gamma;A\vdash\Delta,D,B;C}{\Gamma;\vdash\Delta,D,C,B;D}\,Coup_\mu}{\Gamma;\vdash\Delta,D,C;B}\,Chg$$

Attention au renommage d'une éventuelle formule B déjà présente avec le même nom dans la preuve de $\Gamma;A\vdash\Delta,D;C$ qui est maintenant considérée comme une preuve de $\Gamma;A\vdash\Delta,D,B;C$.

$$\dfrac{\dfrac{\Gamma,C;\vdash\Delta,A;D}{\Gamma;\vdash\Delta,A;C\to D}\,\to_d\quad \Gamma;A\vdash\Delta;B}{\Gamma;\vdash\Delta,B;C\to D}\,Coup_\mu \quad\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\Gamma,C;\vdash\Delta,A;D\quad\Gamma,C;A\vdash\Delta;B}{\Gamma,C;\vdash\Delta,B;D}\,Coup_\mu}{\Gamma;\vdash\Delta,B;C\to D}\,\to_d$$

Attention au renommage éventuel de la formule C déjà présente avec le même nom dans la preuve de $\Gamma;A\vdash\Delta;B$ qui est maintenant considérée comme une preuve de $\Gamma,C;A\vdash\Delta;B$.

$$\dfrac{\dfrac{\Gamma,D;D\vdash\Delta,A;C}{\Gamma,D;\vdash\Delta,A;C}\,Déch\quad \Gamma,D;A\vdash\Delta;B}{\Gamma,D;\vdash\Delta,B;C}\,Coup_\mu \quad\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\Gamma,D;D\vdash\Delta,A;C\quad\Gamma,D;A\vdash\Delta;B}{\Gamma,D;D\vdash\Delta,B;C}\,Coup_\mu}{\Gamma,D;\vdash\Delta,B;C}\,Déch$$

- contrepartie logique de la diffusion d'une liste d'arguments à travers une liste d'arguments

$$\dfrac{\dfrac{\Gamma;\vdash\Delta,A;E\quad\Gamma;D\vdash\Delta,A;C}{\Gamma;E\to D\vdash\Delta,A;C}\,\to_g\quad \Gamma;A\vdash\Delta;B}{\Gamma;E\to D\vdash\Delta,B;C}\,Coup_\mu$$

$$\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\Gamma;\vdash\Delta,A;E\quad\Gamma;A\vdash\Delta;B}{\Gamma;\vdash\Delta,B;E}\,Coup_\mu\quad \dfrac{\Gamma;D\vdash\Delta,A;C\quad\Gamma;A\vdash\Delta;B}{\Gamma;D\vdash\Delta,B;C}\,Coup_\mu}{\Gamma;E\to D\vdash\Delta,B;C}\,\to_g$$

$$\dfrac{\dfrac{}{\Gamma;D\vdash\Delta,A;D}\,Ax\quad \Gamma;A\vdash\Delta;B}{\Gamma;D\vdash\Delta,B;D}\,Coup_\mu \quad\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{}{\Gamma;D\vdash\Delta,B;D}\,Ax$$

- contrepartie logique de la substitution à travers les termes fraîchement nommés

$$\dfrac{\Gamma;\vdash\Delta,B;A \quad \dfrac{\Gamma,A;\vdash\Delta,B,C;B}{\Gamma,A;\vdash\Delta,B;C}\,Chg}{\Gamma;\vdash\Delta,B;C}\,Coup_M \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma;\vdash\Delta,B,C;A \quad \Gamma,A;\vdash\Delta,B,C;B}{\Gamma;\vdash\Delta,B,C;B}\,Coup_M}{\Gamma;\vdash\Delta,B;C}\,Chg$$

Attention au renommage éventuel de la formule C déjà présente avec le même nom dans la preuve de $\Gamma;\vdash\Delta,B;A$ qui est maintenant considérée comme une preuve de $\Gamma;\vdash\Delta,B,C;A$.

Remarque: LKT est essentiellement bilatère et on ne peut lui appliquer directement l'analyse faite en section 1.3. Que peut-on dire cependant de la manière dont sont résolus les conflits ? La réponse est dans la première règle dite “contrepartie logique de la diffusion d'une liste d'arguments à travers un terme”. On y remarque que c'est le côté droit qui fait se dupliquer l'autre, quelle que soit le connecteur de tête de la formule de coupure.

3.3 L'assignement des preuves de LKT par des $\overline{\lambda}\mu$-expressions

Nous donnons un assignement des preuves de LKT par des $\overline{\lambda}\mu$-expressions. Les règles de réduction du $\overline{\lambda}$-calcul et les règles de réécriture pour l'élimination des coupures s'identifieront par cet assignement.

Une **déclaration d'hypothèse** est une autre appellation pour désigner une variable nommée par un nom de variable de terme. La notation pour une déclaration d'hypothèse est $x : A$. Une **déclaration de conclusion** est une autre appellation pour désigner une variable nommée par un nom de μ-variable. La notation pour une déclaration de conclusion est $\alpha : A$.

Nous reprenons le vocable d'**environnement d'hypothèses** pour désigner un ensemble de déclarations d'hypothèses dont aucunes formules ne partagent le même nom de variable. Pareillement pour les environnement de conclusions. Nous reprenons aussi les lettres grecques Γ, Δ, Ω pour désigner ces environnements.

Un **jugement classique** est une expression syntaxique de la forme $\Gamma;\Pi\vdash\Delta;t:A$. Dans cette écriture, Γ est un environnement d'hypothèses, Δ est un environnement de conclusions, Π est un ensemble soit vide soit contenant une formule qui est notée sous la forme d'une déclaration de place (i.e. $.\ : B$), t est une $\overline{\lambda}\mu$-expression et A est une formule.

Comme pour l'assignement de $\overline{\lambda}$-expressions aux preuves de LJT, on assigne les preuves de LKT avec bénitier vide par des $\overline{\lambda}\mu$-termes et les preuves avec bénitier non vide par des listes d'arguments. Là encore, on adopte la notation alternative des listes d'arguments sous forme de **contexte applicatif**.

Règles de formation de contextes applicatifs

Liste vide d'arguments
$$\dfrac{}{\Gamma;.\ :A\vdash\Delta;(.\ [\,]):A}\,Ax$$

Ajout d'un argument à une liste
$$\dfrac{\Gamma;\vdash\Delta;u:A \quad \Gamma;.\ :B\vdash\Delta;(.\ l):C}{\Gamma;.\ :A\to B\vdash\Delta;(.\ [u::l]):C}\,\to_g$$

Concaténation de listes d'arguments
$$\dfrac{\Gamma;.\ :C\vdash\Delta;(.\ l):A \quad \Gamma;.\ :A\vdash\Delta;(.\ l'):B}{\Gamma;.\ :C\vdash\Delta;(.\ (l\ @\ l')):B}\,Coup_T$$

Substitution dans une liste d'arguments
$$\dfrac{\Gamma;\vdash\Delta;u:A \quad \Gamma,x:A;.\ :C\vdash\Delta;(.\ l):B}{\Gamma;.\ :C\vdash\Delta;(.\ l[x:=u]):B}\,Coup_M$$

Diffusion dans une liste d'arguments
$$\dfrac{\Gamma;.\ :D\vdash\Delta,\alpha:A;(.\ l'):C \quad \Gamma;.\ :A\vdash\Delta;(.\ l):B}{\Gamma;.\ :D\vdash\Delta,\alpha:B;(.\ l')[[\alpha]\chi:=[\alpha](\chi\ l)]:C}\,Coup_\mu$$

Règles de formation de termes

Terme applicatif faiblement normal
$$\frac{\Gamma, x:A; .:A \vdash \Delta; (.\ l):B}{\Gamma, x:A; \vdash \Delta; (x\ l):B}\ \textit{Déch}$$

Abstraction
$$\frac{\Gamma, x:A; \vdash \Delta; u:B}{\Gamma; \vdash \Delta; \lambda x.u:A \to B}\ \to_d$$

Terme applicatif non faiblement normal
$$\frac{\Gamma; \vdash \Delta; u:A \quad \Gamma; .:A \vdash \Delta; (.\ l):B}{\Gamma; \vdash \Delta; (u\ l):B}\ Coup_T$$

Substitution dans un terme
$$\frac{\Gamma; \vdash \Delta; u:A \quad \Gamma, x:A; \vdash \Delta; v:B}{\Gamma; \vdash \Delta; v[x:=u]:B}\ Coup_M$$

Diffusion dans un terme
$$\frac{\Gamma; \vdash \Delta, \alpha:A; u:C \quad \Gamma; .:A \vdash \Delta; (.\ l):B}{\Gamma; \vdash \Delta, \alpha:B; u[[\alpha]\chi := [\alpha](\chi\ l)]:C}\ Coup_\mu$$

Changement de conclusion active
$$\frac{\Gamma; \vdash \Delta, \alpha:A, \beta:B; u:A}{\Gamma; \vdash \Delta, \alpha:A; \mu\beta[\alpha]u:B}\ Chg$$

3.4 Le plongement de LK dans LKT

Toute preuve de $\Gamma \vdash \Delta$ dans LK peut être traduite en une preuve de $\Gamma; \vdash \Delta'; A$ dans LKT où la formule active A est une formule de Δ ayant perdu son nom, disons α, et telle que Δ', A^α dénote l'environnement Δ. La traduction est la suivante :

$$\frac{}{\Gamma, A \vdash \Delta, A}\ Ax \qquad \text{se traduit par} \qquad \frac{\frac{}{\Gamma; A \vdash \Delta; A}\ Ax}{\Gamma, A; \vdash \Delta; A}\ \textit{Déch}$$

$$\frac{\begin{array}{c}\vdots\\ \Gamma, A \to B \vdash \Delta, A\end{array} \quad \begin{array}{c}\vdots\\ \Gamma, A \to B, B \vdash \Delta, C\end{array}}{\Gamma, A \to B \vdash \Delta, C}\ \to_g$$

se traduit par

$$\frac{\frac{\frac{\frac{\begin{array}{c}\vdots\\ \Gamma, A \to B; \vdash \Delta'', A; E\end{array}}{\Gamma, A \to B; \vdash \Delta; A}\ Chg \quad \Gamma, A \to B; B \vdash \Delta; B}{\Gamma, A \to B; A \to B \vdash \Delta; B}\ \to_g}{\Gamma, A \to B; \vdash \Delta; B}\ \textit{Déch} \quad \begin{array}{c}\vdots\\ \Gamma, A \to B, B; \vdash \Delta', C; D\end{array}}{\Gamma, A \to B; \vdash \Delta', C; D}\ Coup_M$$

où Δ', D^α et Δ'', E^β sont d'autres représentations de Δ, pour un certain nommage des formules D et E

$$\frac{\begin{array}{c}\vdots\\ \Gamma, A \vdash \Delta, B\end{array}}{\Gamma \vdash \Delta, A \to B}\ \to_d \qquad \text{se traduit par} \qquad \frac{\frac{\begin{array}{c}\vdots\\ \Gamma, A; \vdash \Delta', B; D\end{array}}{\Gamma, A; \vdash \Delta; B}\ Chg}{\Gamma; \vdash \Delta; A \to B}\ \to_d$$

$$\frac{\begin{array}{c}\vdots\\ \Gamma \vdash \Delta, A\end{array} \quad \begin{array}{c}\vdots\\ \Gamma, A \vdash \Delta, B\end{array}}{\Gamma \vdash \Delta, B}\ Coupe \qquad \text{se traduit par} \qquad \frac{\frac{\begin{array}{c}\vdots\\ \Gamma; \vdash \Delta', A; D\end{array}}{\Gamma; \vdash \Delta; A}\ Chg \quad \begin{array}{c}\vdots\\ \Gamma, A; \vdash \Delta'', B; E\end{array}}{\Gamma; \vdash \Delta'', B; E}\ Coup_M$$

où Δ', D^α et Δ'', E^β sont d'autres représentations de Δ, pour un certain nommage des formules D et E

Des simplifications de cette traduction sont possibles. En particulier, la règle d'échange peut s'avérer parfois inutile.

3.5 Termination forte de la réduction en l'un et l'autre calculs

De par l'isomorphisme, la termination forte de l'élimination des coupures pour LKT et la termination forte de la réduction pour les $\overline{\lambda}\mu$-expressions typables sont équivalentes. Nous montrons dans le cadre du $\overline{\lambda}\mu$-calcul ce qui change par rapport à la preuve pour le $\overline{\lambda}$-calcul.

La preuve de termination forte

Nous montrons ce qu'il convient d'ajouter à la preuve de forte normalisation des $\overline{\lambda}$-expressions typables pour obtenir une preuve de forte normalisation des $\overline{\lambda}\mu$-expressions typables selon la nouvelle réduction $\xrightarrow{1}$.

Sur les 8 règles supplémentaires du $\overline{\lambda}\mu$-calcul, 6 sont associées à la règle de μ-coupure, 1 est associée à la règle de coupure de tête et la dernière à la règle de coupure médiane. Il se passe que la règle de μ-coupure se comporte de la même manière que la règle de coupure médiane. Pour étendre la preuve de terminaison de la réduction pour les $\overline{\lambda}$-expressions typables, il suffit donc d'inclure le cas où e a la forme $\mu\beta[\alpha]u$ dans la preuve des lemmes 1 et 2, de prouver le lemme 4 supplémentaire énoncé ci-dessous et d'utiliser ce lemme supplémentaire pour traiter le cas où e a la forme $\mu\beta[\alpha]u$ dans la preuve du lemme 3. La proposition finale utilisera alors les quatre lemmes 1, 2, 3 et 4.

Lemme 4 *Soit e et l des $\overline{\lambda}$-expressions FEN. Si, pour tout u FEN, la typabilité de $(u\ l)$ entraine sa FEN, alors, à son tour, la typabilité de $e[[\alpha]\chi := [\alpha](\chi\ l)]$ entraine sa FEN.*

Nous n'explicitons pas la preuve de ce lemme qui se fait de manière similaire à la preuve du lemme 2.

3.6 Le $\overline{\lambda}_C$-calcul et le calcul LJT + $\neg\neg A \to A$

Un autre cadre pour la logique classique est celui de LJT + élimination de la double négation. Il lui correspond le $\overline{\lambda}$-calcul enrichi de l'opérateur C de Felleisen.

Le $\overline{\lambda}_C$-calcul

L'ensemble des **$\overline{\lambda}_C$-expressions** est défini par

$$\begin{array}{lrcl} \textbf{Termes:} & t & ::= & (p\ l) \mid (\lambda t) \mid (t\ l) \mid (t[p := t]) \mid C(t) \\ \textbf{Listes d'arguments:} & l & ::= & [\,] \mid [t :: l] \mid [l\ @\ l] \mid l[p := t] \end{array}$$

L'ensemble des $\overline{\lambda}_C$-expressions **normales** est défini par

$$\begin{array}{rcl} t & ::= & (p\ l) \mid (\lambda t) \mid C(t) \\ l & ::= & [\,] \mid [t :: l] \end{array}$$

Par rapport au $\overline{\lambda}$-calcul, on considère les règles de réduction supplémentaires suivantes :

- C-réduction

$$(C(u)\ l) \quad \xrightarrow{r} \quad C(\lambda k.(u\ [\lambda x.(k\ [(x\ l)])]))$$

où k et x sont des noms n'apparaissant pas déjà dans u et l

- règle supplémentaire de propagation de la substitution

$$C(u)[x := v] \quad \xrightarrow{r} \quad C(u[x := v])$$

Le calcul LJT + $\neg\neg A \to A$

Comme pour LKT, on considère l'ensemble des formules formées à partir de la constante supplémentaire $\perp$. Notation : $\neg A$ est une abréviation pour la formule $A \to \perp$.

On ajoute donc à LJT la règle

$$\frac{\Gamma; \vdash \neg\neg A}{\Gamma; \vdash A}$$

avec l'assignement

$$\frac{\Gamma; \vdash u : \neg\neg A}{\Gamma; \vdash C(u) : A}$$

ce qui permet de retrouver la logique classique.

Chapitre 4

Autres extensions du $\overline{\lambda}$-calcul

4.1 Extension de LJT à la disjonction et du $\overline{\lambda}$-calcul aux injections et à l'opérateur de choix

4.1.1 Le $\overline{\lambda}$-calcul avec injections et opérateur de choix

Le nouvel ensemble de $\overline{\lambda}$-expressions est défini par :

$$\begin{array}{lll} \textbf{Termes :} & t & ::= \ (x\ l) \mid (\lambda x.t) \mid (t\ l) \mid (t[x := t]) \mid inj_1(t) \mid inj_2(t) \\ \textbf{Listes d'arguments :} & l & ::= \ [\,] \mid [t :: l] \mid (l\ @\ l) \mid l[x := t] \mid [<(x)t|(x)t>] \end{array}$$

La liste d'argument $[<(x)t|(y)u>]$ joue le rôle d'un opérateur de choix. Si on la regarde comme un contexte applicatif $(.\ [<(x)t|(y)u>])$ alors on obtient une notation plus expressive en écrivant :

$$case\ .\ \ of\ inj_1(x) \ {-}{>}\ t \mid inj_2(y) \ {-}{>}\ u$$

Le sous-ensemble des formes normales est défini par :

$$\begin{array}{lll} t & ::= & (x\ l) \mid (\lambda x.t) \mid inj_1(t) \mid inj_2(t) \\ l & ::= & [\,] \mid [t :: l] \mid [<(x)t|(x)t>] \end{array}$$

Règles de réduction

Aux règles de réduction de la version minimale du $\overline{\lambda}$-calcul, on ajoute les règles de réduction suivantes :

- règles de réduction de l'opérateur de choix

$$\begin{array}{rcl} (inj_1(t)\ [<(x)u|(y)v>] & \xrightarrow{r} & u[x := t] \\ (inj_2(t)\ [<(x)u|(y)v>] & \xrightarrow{r} & v[y := t] \end{array}$$

- règles de réduction supplémentaire pour la propagation de la substitution

$$\begin{array}{rcl} inj_1(t)[z := v] & \xrightarrow{r} & inj_1(t[z := v]) \\ inj_2(t)[z := v] & \xrightarrow{r} & inj_2(t[z := v]) \\ [<(x)u|(y)v>][z := w] & \xrightarrow{r} & [<(x)u[z := w]|(y)v[z := w]>] \end{array}$$

  attention à une possible capture de variable dans la troisième règle

- règle de réduction supplémentaire pour le calcul de la concaténation

$$[<(x)u|(y)v>]\ @\ l' \ \xrightarrow{r}\ [<(x)(u\ l')|(y)(v\ l')>]$$

Remarque : La règle

$$((case\ u\ of\ inj_1(x) \ {-}{>}\ v \mid inj_2(y) \ {-}{>}\ w)\ l) \xrightarrow{r} (case\ u\ of\ inj_1(x) \ {-}{>}\ (v\ l) \mid inj_2(y) \ {-}{>}\ (w\ l))$$

est un cas particulier de la règle de calcul de la concaténation en présence de l'opérateur de choix. On n'a pas cette spécificité de la disjonction comme il y a dans le λ-calcul habituel.

4.1.2 Le calcul LJT avec $\vee$

On ajoute à LJT les règles suivantes d'introduction de la disjonction :

$$\frac{\Gamma; \vdash A}{\Gamma; \vdash A \vee B}\ \vee_d^1 \qquad \frac{\Gamma; \vdash B}{\Gamma; \vdash A \vee B}\ \vee_d^2$$

$$\frac{\Gamma, A; \vdash C \quad \Gamma, B; \vdash C}{\Gamma; A \vee B \vdash C}\ \vee_g$$

4.1.3 L'assignement par des $\overline{\lambda}$-expressions

$$\frac{\Gamma; \vdash u : A}{\Gamma; \vdash inj_2(u) : A \vee B}\ \vee_d^1 \qquad \frac{\Gamma; \vdash v : B}{\Gamma; \vdash inj_2(v) : A \vee B}\ \vee_d^2$$

$$\frac{\Gamma, x : A; \vdash u : C \qquad \Gamma, y : B; \vdash v : C}{\Gamma; .\ : A \vee B \vdash (.\ \ [<(x)u|(y)v>]) : C}\ \vee_g$$

Pour éliminer les coupures de manière exhaustive, il faut considérer quelques règles supplémentaires d'élimination. Ce sont les mêmes, version typée, que pour le $\overline{\lambda}$-calcul avec injection et opérateur de choix.

4.2 Extension de LJT à la conjonction et du $\overline{\lambda}$-calcul à la paire

L'extension de l'isomorphisme à la conjonction est simple. La règle d'introduction droite de la conjonction est interprétée par un constructeur de paire. Deux sortes de règles d'introduction gauche de la conjonction sont envisageables. L'une est interprétée par les projections et l'autre par l'opérateur de décomposition de la paire (du style de celui présent dans la théorie des types de Martin-Löf [42]).

4.2.1 Le $\overline{\lambda}$-calcul avec constructeur de paire, projections et opérateur de décomposition de la paire

Le nouvel ensemble de $\overline{\lambda}$-expressions est défini par :

Termes :	t	$::=$	$(x\ l) \mid (\lambda x.t) \mid (t\ l) \mid (t[x := t]) \mid\ <t, t>$
Listes d'argument :	l	$::=$	$[\ ] \mid [t :: l] \mid (l\ @\ l) \mid l[x := t] \mid [<x, x> t] \mid [\pi_1 :: l] \mid [\pi_2 :: l]$

Le sous-ensemble des formes normales est défini par :

$$\begin{array}{lcl} t & ::= & (x\ l) \mid (\lambda x.t) \mid\ <t, t> \\ l & ::= & [\ ] \mid [t :: l] \mid [<x, x> t] \mid [\pi_1 :: l] \mid [\pi_2 :: l] \end{array}$$

Remarque : La présence simultanée des projections et de l'opérateur de décomposition de la paire est redondante relativement à la complétude du calcul intuitionniste avec conjonction.

Règles de réduction

Aux règles de réduction de la version minimale du $\overline{\lambda}$-calcul, on ajoute les règles de réduction suivantes :

- règles de destruction de la paire

$$\begin{array}{rcl} (<t, u>\ [\pi_1 :: l]) & \xrightarrow{r} & (t\ l) \\ (<t, u>\ [\pi_2 :: l]) & \xrightarrow{r} & (u\ l) \\ (<t, u>\ [<x, y>\ v]) & \xrightarrow{r} & v[x := t][y := u] \end{array}$$

- règles de réduction supplémentaire pour la propagation de la substitution

$$\begin{array}{rcl} <t,u>[z:=v] & \xrightarrow{r} & <t[z:=v],u[z:=v]> \\ [\pi_1 :: l][z:=v] & \xrightarrow{r} & [\pi_1 :: l[z:=v]] \\ [\pi_2 :: l][z:=v] & \xrightarrow{r} & [\pi_2 :: l[z:=v]] \\ [<x,y>\, v][z:=v] & \xrightarrow{r} & [<x,y>\,(v[z:=v])] \end{array}$$

attention à une possible capture de variable dans la quatrième règle

- règles de réduction supplémentaire pour le calcul de concaténation

$$\begin{array}{rcl} [\pi_1 :: l] \;@\; l' & \xrightarrow{r} & [\pi_1 :: (l \;@\; l')] \\ [\pi_2 :: l] \;@\; l' & \xrightarrow{r} & [\pi_2 :: (l \;@\; l')] \\ [<x,y>\, v] \;@\; l' & \xrightarrow{r} & [<x,y>\,(v\; l')] \end{array}$$

4.2.2 Le calcul LJT avec $\wedge$

On ajoute à LJT les règles suivantes d'introduction de la conjonction :

$$\frac{\Gamma; \vdash A \quad \Gamma; \vdash B}{\Gamma; \vdash A \wedge B}\ \wedge_d$$

$$\frac{\Gamma; A \vdash C}{\Gamma; A \wedge B \vdash C}\ \wedge_g^1 \qquad \frac{\Gamma; B \vdash C}{\Gamma; A \wedge B \vdash C}\ \wedge_g^2$$

$$\frac{\Gamma, A, B; \vdash C}{\Gamma; A \wedge B \vdash C}\ \wedge_g^{sym}$$

4.2.3 L'assignement par des $\overline{\lambda}$-expressions

$$\frac{\Gamma; \vdash u : A \quad \Gamma; \vdash v : B}{\Gamma; \vdash <u,v>: A \wedge B}\ \wedge_d$$

$$\frac{\Gamma; .:A \vdash (.\ \ l) : C}{\Gamma; .:A \wedge B \vdash \Delta; (.\ \ [\pi_1 :: l]) : C}\ \wedge_g^1 \qquad \frac{\Gamma; .:B \vdash (.\ \ l) : C}{\Gamma; .:A \wedge B \vdash \Delta; (.\ \ [\pi_2 :: l]) : C}\ \wedge_g^2$$

$$\frac{\Gamma, x:A, y:B; \vdash u : C}{\Gamma; .:A \wedge B \vdash \Delta; (.\ \ [<x,y>\, u]) : C}\ \wedge_g^{sym}$$

Pour éliminer les coupures de manière exhaustive, il faut considérer quelques règles supplémentaires d'élimination. Ce sont les mêmes, version typée, que pour le $\overline{\lambda}$-calcul avec paire, projections et opérateur de décomposition de la paire.

4.3 Extension de la correspondance pour LJT avec quantificateurs

On considère une version de LJT avec quantification sur des objets du premier ordre. Pour cela, on se donne, pour tout n, un ensemble $\mathcal{V}_O^n$ de variables de fonctions d'arité n (ou d'individus si $n = 0$). On note r, s, ... les variables d'individus.

On définit l'ensemble des individus par la règle de grammaire $a ::= r(a, ..., a)$ dans laquelle r parcourt les $\mathcal{V}_O^n$ et si r est dans $\mathcal{V}_O^n$ alors $(a, ..., a)$ est un n-uplet. On utilise les lettres a, b, c, ... pour désigner les individus.

4.3.1 Le calcul LJT avec quantificateurs

On se donne une famille $\mathcal{V}_F^n$ d'ensembles de variables propositionnelles (telle que $\mathcal{V}_F^0 = \mathcal{V}_F$).

Les formules sont définies par la grammaire suivante

$$A \quad ::= \quad P(a, ..., a) \mid A \to A \mid \forall x.A \mid \exists x.A$$

où x parcourt $\mathcal{V}_C$ et P parcourt les $\mathcal{V}_F^n$ de telle manière que si P est dans $\mathcal{V}_F^n$, alors $(a, ..., a)$ est un n-uplet d'individus.

Les règles de LJT sont celles du fragment minimal de LJT auxquelles il faut ajouter les règles suivantes d'introduction des quantificateurs :

$$\frac{\Gamma; A\{r := a\} \vdash B}{\Gamma; \forall r.A \vdash B}\ \forall_g \qquad \frac{\Gamma; \vdash A}{\Gamma; \vdash \forall r.A}\ \forall_d$$

$$\frac{\Gamma, A; \vdash B}{\Gamma; \exists r.A \vdash B}\ \exists_g \qquad \frac{\Gamma; \vdash A\{r := a\}}{\Gamma; \vdash \exists r.A}\ \exists_d$$

où $A\{r := a\}$ désigne la formule A dans laquelle toute occurrence de r a été remplacée par une occurrence de a. Aussi, dans les règles $\forall_d$ et $\exists_g$, la variable d'individu r ne doit pas figurer dans les formules de Γ.

4.3.2 Le $\overline{\lambda}$-calcul correspondant

L'assignement des preuves de LJT avec quantificateurs se fait par des $\overline{\lambda}$-expressions avec paire et destructeur de paire. La définition de ces $\overline{\lambda}$-expressions est la suivante :

$$\begin{array}{llll} \textbf{Termes :} & t & ::= & (x\ l) \mid (\lambda x.t) \mid (t\ l) \mid (t[x := t]) \\ & & & \mid \lambda s.t \mid\ <a, t> \\ \textbf{Listes d'argument :} & l & ::= & [\,] \mid [t :: l] \mid (l\ @\ l) \mid l[x := t] \\ & & & \mid [a :: l] \mid [<r, x>\ t] \end{array}$$

où s parcourt l'ensemble des variables d'individu et a parcourt l'ensemble des individus.

Le sous-ensemble des formes normales est défini par :

$$\begin{array}{lll} t & ::= & (x\ l) \mid (\lambda x.t) \mid \lambda s.t \mid\ <a, t> \\ l & ::= & [\,] \mid [t :: l] \mid [a :: l] \mid [<r, x>\ t] \end{array}$$

Règles de réduction

Les règles de réduction sont celles du $\overline{\lambda}$-calcul avec paire et destructeur de paire, à ceci près que la substitution par des individus est effectuée d'un coup.

Les règles à modifier sont les suivantes :

$$\begin{array}{rcl} (\lambda r.u\ [a :: l]) & \xrightarrow{r} & (u\{r := a\}\ l) \\ (<a, t>\ [<r, x>\ u]) & \xrightarrow{r} & u\{r := a\}[x := t] \end{array}$$

où $u\{r := a\}$ représente u dans lequel toutes les occurrences de r ont été remplacées par a. Attention néanmoins aux possibles captures de variables d'individu.

4.3.3 L'assignement par des $\overline{\lambda}$-expressions

$$\frac{\Gamma;.:A\{x:=t\}\vdash(.\ l):B}{\Gamma;.:\forall x.A\vdash\Delta;(.\ [t::l]):B}\ \forall_g \qquad \frac{\Gamma;\vdash u:A}{\Gamma;\vdash\lambda x.u:\forall x.A}\ \forall_d$$

$$\frac{\Gamma,y:A;\vdash u:B}{\Gamma;.:\exists x.A\vdash(.\ [<x,y>u]):B}\ \exists_g \quad \frac{\Gamma;\vdash u:A\{x:=t\}}{\Gamma;\vdash<t,u>:\exists x.A}\ \exists_d$$

Les règles d'élimination des coupures se déduisent, mais en y intégrant les contraintes de types, de celles du $\overline{\lambda}$-calcul décrit dans la section précédente.

4.4 La correspondance pour le calcul classique des prédicats

Les extensions précédentes peuvent être rassemblées de manière à obtenir un isomorphisme pour le calcul classique des prédicats au grand complet. On peut aussi former une variante avec indices de de Bruijn étendant celle décrite ci-après.

4.5 L'isomorphisme pour le calcul classique avec indices de de Bruijn

4.5.1 Le $\overline{\lambda}\mu$-calcul avec indices de de Bruijn

Un intérêt pour présenter une version du $\overline{\lambda}\mu$-calcul (et donc du $\overline{\lambda}$-calcul) avec indices de de Bruijn est de souligner la correspondance entre la règle d'affaiblissement du calcul des séquents et l'opérateur de décalage vers le haut (le "lift operator" des anglophones) des λ-calculs avec indices de de Bruijn. De plus, la réduction s'exprime avec plus de précision dans ce cadre (pas de raisonnement à α-équivalence près).

Les $\overline{\lambda}\mu$-expressions avec indices de de Bruijn

L'ensemble des $\overline{\lambda}\mu$-**expressions**, recouvrant les $\overline{\lambda}\mu$-**termes** et les **listes d'arguments** est défini inductivement par la grammaire suivante où p parcourt l'ensemble des entiers naturels :

$$\begin{array}{llll} \textbf{Termes :} & t & ::= & (p\ l) \mid (\lambda t) \mid (t\ l) \mid (t[p:=t]) \mid (t{\uparrow_p}) \\ & & & \mu[p]t \mid t[[p]\chi:=[p](\chi\ l)] \mid (t{\uparrow^p}) \\ \textbf{Listes d'arguments :} & l & ::= & [\,] \mid [t::l] \mid [l \mathbin{@} l] \mid l[p:=t] \mid (l{\uparrow_p}) \\ & & & l[[p]\chi:=[p](\chi\ l)] \mid (l{\uparrow^p}) \end{array}$$

Les entiers figurant dans les $\overline{\lambda}\mu$-expressions sont appelés indices de variables de terme ou indices de μ-variables, selon qu'ils occupent la place d'une variable de terme ou d'une μ-variable.

Le terme $t{\uparrow_p}$ représente l'opération explicite de décalage vers le haut d'une unité des indices de variables de termes plus grands que p dans le terme t. Pareillement pour les listes d'arguments.

Le terme $t{\uparrow^p}$ représente l'opération explicite de décalage vers le haut d'une unité des indices de μ-variables plus grands que p dans le terme t. Pareillement pour les listes d'arguments.

Remarque : χ est ici un symbole spécial pour la "lisibilité" des notations.

Formes normales du $\overline{\lambda}\mu$-calcul avec indices de de Bruijn

Une $\overline{\lambda}\mu$-expression est **normale** si elle ne contient pas d'opérateurs de concaténation, de substitution ou de décalage vers le haut et si les sous-expressions de la forme $(t\ l)$ sont exclues.

Autrement dit, une $\overline{\lambda}\mu$-expression est normale si elle est formée à partir de la grammaire suivante :

$$\begin{array}{lll} t & ::= & (p\ l) \mid (\lambda t) \mid \mu[p]t \\ l & ::= & [\,] \mid [t::l] \end{array}$$

Règles de réduction du $\overline{\lambda}\mu$-calcul avec indices de de Bruijn

Les règles de réduction du $\overline{\lambda}\mu$-calcul avec indices de de Bruijn correspondent à peu près à celles du $\overline{\lambda}\mu$-calcul avec liaisons par noms. On y trouve en plus un lot de règles pour le calcul des opérateurs de décalage vers le haut.

- β-réduction

$$\begin{array}{rcll} (\lambda u\ [v :: l]) & \stackrel{r}{\to} & (u[0 := v] :: l) & \beta_{cons} \\ (\lambda u\ [\,]) & \stackrel{r}{\to} & \lambda u & \beta_{nil} \end{array}$$

- μ-réduction

$$(\mu[p]u\ l) \stackrel{r}{\to} \mu[p](u[[0]\chi := [0](\chi\ l)]) \quad \mu$$

- indication de concaténation

$$((p\ l)\ l') \stackrel{r}{\to} (p\ (l\ @\ l')) \quad C_{var}$$

- règles de calcul de la concaténation

$$\begin{array}{rcll} [u :: l]\ @\ l' & \stackrel{r}{\to} & [u :: (l\ @\ l')] & C_{cons} \\ [\,]\ @\ l' & \stackrel{r}{\to} & l' & C_{nil} \end{array}$$

- règles de propagation de la substitution à travers les termes faiblement normaux

$$\begin{array}{rcll} (p\ l)[p := v] & \stackrel{r}{\to} & (v\ l[p := v]) & S_{yes} \\ (q\ l)[p := v] & \stackrel{r}{\to} & (q\ l[p := v]) & S_{>}\ (p > q) \\ (q\ l)[p := v] & \stackrel{r}{\to} & (q-1\ l[p := v]) & S_{<}\ (p < q) \\ (\lambda u)[p := v] & \stackrel{r}{\to} & \lambda(u[p+1 := v\uparrow_0]) & S_{\lambda} \\ (\mu[p]u)[q := v] & \stackrel{r}{\to} & \mu[p](u[q := v\uparrow^0]) & S_{\mu} \end{array}$$

- règles de propagation de la substitution à travers les listes d'arguments faiblement normales

$$\begin{array}{rcll} [\,][p := v] & \stackrel{r}{\to} & [\,] & S_{nil} \\ [u :: l][p := v] & \stackrel{r}{\to} & [u[p := v] :: l[p := v]] & S_{cons} \end{array}$$

- règles de diffusion d'une liste d'arguments à travers un terme

$$\begin{array}{rcll} (\mu[p]u)[[p]\chi := [p](\chi\ l)] & \stackrel{r}{\to} & \mu[p](u[[p+1]\chi := [p+1](\chi\ l\uparrow^0)]\ l\uparrow^p\uparrow^0) & D_{yes} \\ (\mu[q]u)[[p]\chi := [p](\chi\ l)] & \stackrel{r}{\to} & \mu[q](u[[p+1]\chi := [p+1](\chi\ l\uparrow^0)]) & D_{no}\ (p \neq q) \\ (\lambda u)[[p]\chi := [p](\chi\ l)] & \stackrel{r}{\to} & \lambda(u[[p]\chi := [p](\chi\ l\uparrow_0)]) & D_{\lambda} \\ (q\ l')[[p]\chi := [p](\chi\ l)] & \stackrel{r}{\to} & (q\ (l'[[p]\chi := [p](\chi\ l)])) & D_{app} \end{array}$$

- règles de diffusion d'une liste d'arguments à travers une liste d'arguments

$$\begin{array}{rcll} ([u :: l'])[[p]\chi := [p](\chi\ l)] & \stackrel{r}{\to} & [u[[p]\chi := [p](\chi\ l)] :: l'[[p]\chi := [p](\chi\ l)]] & D_{cons} \\ ([\,])[[p]\chi := [p](\chi\ l)] & \stackrel{r}{\to} & [\,] & D_{nil} \end{array}$$

- règles de calcul de l'opérateur de décalage vers le haut des variables de termes à travers les termes faiblement normaux

$$
\begin{array}{rcll}
(\lambda u)\uparrow_p & \xrightarrow{r} & \lambda(u\uparrow_{p+1}) & L_\lambda \\
(q\ l)\uparrow_p & \xrightarrow{r} & (q{+}1\ (l\uparrow_p)) & L_{yes}\ (q \geq p) \\
(q\ l)\uparrow_p & \xrightarrow{r} & (q\ (l\uparrow_p)) & L_{no}\ (q < p) \\
(\mu[q]u)\uparrow_p & \xrightarrow{r} & \mu[q](u\uparrow_p) & L_\mu
\end{array}
$$

- règles de calcul de l'opérateur de décalage vers le haut des variables de termes à travers les listes d'arguments faiblement normales

$$
\begin{array}{rcll}
[\,]\uparrow_p & \xrightarrow{r} & [\,] & L_{nil} \\
[u :: l]\uparrow_p & \xrightarrow{r} & [u\uparrow_p :: l\uparrow_p] & L_{cons}
\end{array}
$$

- règles de calcul de l'opérateur de décalage vers le haut des μ-variables à travers les termes faiblement normaux

$$
\begin{array}{rcll}
(\lambda u)\uparrow^p & \xrightarrow{r} & \lambda(u\uparrow^p) & L^\lambda \\
(q\ l)\uparrow^p & \xrightarrow{r} & (q\ (l\uparrow^p)) & L^{var} \\
(\mu[q]u)\uparrow^p & \xrightarrow{r} & \mu[q{+}1](u\uparrow^{p+1}) & L^{yes}\ (q \geq p) \\
(\mu[q]u)\uparrow^p & \xrightarrow{r} & \mu[q](u\uparrow^{p+1}) & L^{no}\ (q < p)
\end{array}
$$

- règles de calcul de l'opérateur de décalage vers le haut des μ-variables à travers les listes d'arguments faiblement normales

$$
\begin{array}{rcll}
[\,]\uparrow^p & \xrightarrow{r} & [\,] & L^{nil} \\
[u :: l]\uparrow^p & \xrightarrow{r} & [u\uparrow^p :: l\uparrow^p] & L^{cons}
\end{array}
$$

4.5.2 Le calcul LKT avec règles d'affaiblissement

Environnements comme suites de formules

La définition pour les environnements, appropriée aux indices de de Bruijn, est une définition des environnements comme suites de formules.

Un **environnement** est une suite de formules. Il peut y avoir plusieurs fois la même formule dans la suite. On continue à utiliser les lettres Γ, Δ, Ω,... pour désigner les environnements. Une notation comme Γ, Δ désigne maintenant la juxtaposition de Γ et de Δ.

Un **séquent** a la forme $\Gamma; \Pi \vdash \Delta; A$ où Γ et Δ sont des environnements.

Les règles d'inférence

Les règles d'inférence du calcul LKT avec règles explicites d'affaiblissement sont les suivantes :

règle d'axiome

$$\frac{}{\Gamma; A \vdash \Lambda; A}\ Ax$$

règle de déchargement

$$\frac{\Gamma, A, \Delta; A \vdash \Lambda; B}{\Gamma, A, \Delta; \vdash \Lambda; B}\ D\acute{e}ch$$

règle d'introduction gauche de $\rightarrow$

$$\frac{\Gamma; \vdash \Lambda; A \quad \Gamma; B \vdash \Lambda; C}{\Gamma; A \rightarrow B \vdash \Lambda; C} \rightarrow_g$$

règle d'introduction droite de $\rightarrow$

$$\frac{\Gamma, A; \vdash \Lambda; B}{\Gamma; \vdash A \rightarrow \Lambda; B} \rightarrow_d$$

règle de coupure de tête

$$\frac{\Gamma; \Pi \vdash \Lambda; A \quad \Gamma; A \vdash \Lambda; B}{\Gamma; \Pi \vdash \Lambda; B}\ Coup_T$$

règle de coupure médiane

$$\frac{\Gamma, \Delta; \vdash \Lambda; A \quad \Gamma, A, \Delta; \Pi \vdash \Lambda; B}{\Gamma, \Delta; \Pi \vdash \Lambda; B}\ Coup_M$$

règle de changement de conclusion

$$\frac{\Gamma; \vdash \Lambda, A, \Omega, B; A}{\Gamma; \vdash \Lambda, A, \Omega; B}\ Chg$$

règle de μ-coupure

$$\frac{\Gamma; \Pi \vdash \Lambda, A, \Omega; C \quad \Gamma; A \vdash \Lambda, \Omega; B}{\Gamma; \Pi \vdash \Lambda, B, \Omega; C}\ Coup_\mu$$

règle d'affaiblissement gauche

$$\frac{\Gamma, \Delta; \Pi \vdash \Lambda; B}{\Gamma, A, \Delta; \Pi \vdash \Lambda; B}\ Aff_g$$

règle d'affaiblissement droite

$$\frac{\Gamma; \Pi \vdash \Lambda, \Omega; B}{\Gamma; \Pi \vdash \Lambda, A, \Omega; B}\ Aff_d$$

4.5.3 Elimination des coupures

L'élimination des coupures inclut aussi maintenant l'élimination des règles d'affaiblissement. Ceci excepté, le système de réécriture permettant d'éliminer les coupures n'est pas très différent de celui pour le calcul sans règle d'affaiblissement.

Réduction de la règle $Coup_T$

- contrepartie logique de la β-réduction

$$\frac{\dfrac{\Gamma, A; \vdash \Lambda; B}{\Gamma; \vdash \Lambda; A \rightarrow B} \rightarrow_d \quad \dfrac{\Gamma; \vdash \Lambda; A \quad \Gamma; B \vdash \Lambda; C}{\Gamma; A \rightarrow B \vdash \Lambda; C} \rightarrow_g}{\Gamma; \vdash \Lambda; C}\ Coup_T \quad \stackrel{\text{LKT}}{\longrightarrow} \quad \frac{\dfrac{\Gamma; \vdash \Lambda; A \quad \Gamma, A; \vdash \Lambda; B}{\Gamma; \vdash \Lambda; B}\ Coup_M \quad \Gamma; B \vdash \Lambda; C}{\Gamma; \vdash \Lambda; C}\ Coup_T$$

$$\frac{\dfrac{\Gamma, A; \vdash \Lambda; B}{\Gamma; \vdash \Lambda; A \rightarrow B} \rightarrow_d \quad \dfrac{}{\Gamma; A \rightarrow B \vdash \Lambda; A \rightarrow B}\ Ax}{\Gamma; \vdash \Lambda; A \rightarrow B}\ Coup_T \quad \stackrel{\text{LKT}}{\longrightarrow} \quad \frac{\Gamma, A; \vdash \Lambda; B}{\Gamma; \vdash \Lambda; A \rightarrow B} \rightarrow_d$$

- contrepartie logique de la μ-réduction

$$\frac{\dfrac{\Gamma; \vdash \Lambda, A, \Omega, B; A}{\Gamma; \vdash \Lambda, A, \Omega; B}\ Chg \quad \Gamma; B \vdash \Lambda, A, \Omega; C}{\Gamma; \vdash \Lambda, A, \Omega; C}\ Coup_T \quad \stackrel{\text{LKT}}{\longrightarrow} \quad \frac{\dfrac{\Gamma; \vdash \Lambda, A, \Omega, B; A \quad \Gamma; B \vdash \Lambda, A, \Omega; C}{\Gamma; \vdash \Lambda, A, \Omega, C; A}\ Coup_\mu}{\Gamma; \vdash \Lambda, A, \Omega; C}\ Chg$$

- contrepartie logique de l'indication de concaténation

$$\dfrac{\dfrac{\Gamma, B, \Delta; B \vdash \Lambda; A}{\Gamma, B, \Delta; \vdash \Lambda; A}\,\textit{Déch} \quad \Gamma, B, \Delta; A \vdash \Lambda; C}{\Gamma, B, \Delta; \vdash \Lambda; C}\,\mathit{Coup}_T \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma, B, \Delta; B \vdash \Lambda; A \quad \Gamma, B, \Delta; A \vdash \Lambda; C}{\Gamma, B, \Delta; B \vdash \Lambda; C}\,\mathit{Coup}_T}{\Gamma, B, \Delta; \vdash \Lambda; C}\,\textit{Déch}$$

- contrepartie logique du calcul de la concaténation

$$\dfrac{\dfrac{\Gamma; \vdash \Lambda; D \quad \Gamma; B \vdash \Lambda; A}{\Gamma; D \to B \vdash \Lambda; A}\,\to_g \quad \Gamma; A \vdash \Lambda; C}{\Gamma; D \to B \vdash \Lambda; C}\,\mathit{Coup}_T \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\Gamma; \vdash \Lambda; D \quad \dfrac{\Gamma; B \vdash \Lambda; A \quad \Gamma; A \vdash \Lambda; C}{\Gamma; B \vdash \Lambda; C}\,\mathit{Coup}_T}{\Gamma; D \to B \vdash \Lambda; C}\,\to_g$$

$$\dfrac{\dfrac{}{\Gamma; A \vdash \Lambda; A}\,\mathit{Ax} \quad \Gamma; A \vdash \Lambda; C}{\Gamma; A \vdash \Lambda; C}\,\mathit{Coup}_T \quad \overset{\text{LKT}}{\longrightarrow} \quad \Gamma; A \vdash \Lambda; C$$

Réduction de la règle Coup_M

- contrepartie logique de la propagation de la substitution à travers les termes faiblement normaux

$$\dfrac{\Gamma, \Delta; \vdash \Lambda; A \quad \dfrac{\Gamma, A, \Delta; A \vdash \Lambda; C}{\Gamma, A, \Delta; \vdash \Lambda; C}\,\textit{Déch}}{\Gamma, \Delta; \vdash \Lambda; C}\,\mathit{Coup}_M \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\Gamma, \Delta; \vdash \Lambda; A \quad \dfrac{\Gamma, \Delta; \vdash \Lambda; A \quad \Gamma, A, \Delta; A \vdash \Lambda; C}{\Gamma, \Delta; A \vdash \Lambda; C}\,\mathit{Coup}_M}{\Gamma, \Delta; \vdash \Lambda; C}\,\mathit{Coup}_T$$

$$\dfrac{\Gamma, \Delta, B, \Phi; \vdash \Lambda; A \quad \dfrac{\Gamma, A, \Delta, B, \Phi; B \vdash \Lambda; C}{\Gamma, A, \Delta, B, \Phi; \vdash \Lambda; C}\,\textit{Déch}}{\Gamma, \Delta, B, \Phi; \vdash \Lambda; C}\,\mathit{Coup}_M \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma, \Delta, B, \Phi; \vdash \Lambda; A \quad \Gamma, A, \Delta, B, \Phi; B \vdash \Lambda; C}{\Gamma, \Delta, B, \Phi; B \vdash \Lambda; C}\,\mathit{Coup}_M}{\Gamma, \Delta, B, \Phi; \vdash \Lambda; C}\,\textit{Déch}$$

et une règle similaire avec les places de A et B interverties.

$$\dfrac{\Gamma, \Delta; \vdash \Lambda; A \quad \dfrac{\Gamma, A, \Delta, B; \vdash \Lambda; C}{\Gamma, A, \Delta; \vdash \Lambda; B \to C}\,\to_d}{\Gamma, \Delta; \vdash \Lambda; B \to C}\,\mathit{Coup}_M \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\dfrac{\Gamma, \Delta; \vdash \Lambda; A}{\Gamma, \Delta, B; \vdash \Lambda; A}\,\mathit{Aff}_g \quad \Gamma, A, \Delta, B; \vdash \Lambda; C}{\Gamma, \Delta, B; \vdash \Lambda; C}\,\mathit{Coup}_M}{\Gamma, \Delta; \vdash \Lambda; B \to C}\,\to_d$$

$$\dfrac{\Gamma, \Delta; \vdash \Lambda, B, \Omega; A \quad \dfrac{\Gamma, A, \Delta; \vdash \Lambda, B, \Omega, C; B}{\Gamma, A, \Delta; \vdash \Lambda, B, \Omega; C}\,\mathit{Chg}}{\Gamma, \Delta; \vdash \Lambda, B, \Omega; C}\,\mathit{Coup}_M \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\dfrac{\Gamma, \Delta; \vdash \Lambda, B, \Omega; A}{\Gamma, \Delta; \vdash \Lambda, B, \Omega, C; A}\,\mathit{Aff}_d \quad \Gamma, A, \Delta; \vdash \Lambda, B, \Omega, C; B}{\Gamma, \Delta; \vdash \Lambda, B, \Omega, C; B}\,\mathit{Coup}_M}{\Gamma, \Delta; \vdash \Lambda, B, \Omega; C}\,\mathit{Chg}$$

- contrepartie logique de la propagation de la substitution à travers les listes d'arguments faiblement normales

$$\dfrac{\Gamma,\Delta;\vdash\Lambda;A \quad \dfrac{\Gamma,A,\Delta;\vdash\Lambda;B \quad \Gamma,A,\Delta;C\vdash\Lambda;D}{\Gamma,A,\Delta;B\to C\vdash\Lambda;D}\to_g}{\Gamma,\Delta;B\to C\vdash\Lambda;D}\mathit{Coup}_M$$

$$\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\Gamma,\Delta;\vdash\Lambda;A \quad \Gamma,A,\Delta;\vdash\Lambda;B}{\Gamma,\Delta;\vdash\Lambda;B}\mathit{Coup}_M \quad \dfrac{\Gamma,\Delta;\vdash\Lambda;A \quad \Gamma,A,\Delta;C\vdash\Lambda;D}{\Gamma,\Delta;C\vdash\Lambda;D}\mathit{Coup}_M}{\Gamma,\Delta;B\to C\vdash\Lambda;D}\to_g$$

$$\dfrac{\Gamma,\Delta;\vdash\Lambda;A \quad \dfrac{}{\Gamma,A,\Delta;B\vdash\Lambda;B}\mathit{Ax}}{\Gamma,\Delta;B\vdash\Lambda;B}\mathit{Coup}_M \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{}{\Gamma,\Delta;B\vdash\Lambda;B}\mathit{Ax}$$

Réduction de la règle Coup_μ

- contrepartie logique de la diffusion d'une liste d'arguments à travers un terme

$$\dfrac{\dfrac{\Gamma;\vdash\Lambda,A,\Omega,B;A}{\Gamma;\vdash\Lambda,A,\Omega;B}\mathit{Chg} \quad \Gamma;A\vdash\Lambda,\Omega;C}{\Gamma;\vdash\Lambda,C,\Omega;B}\mathit{Coup}_\mu$$

$$\overset{\text{LKT}}{\longrightarrow}\quad \dfrac{\dfrac{\dfrac{\Gamma;\vdash\Lambda,A,\Omega,B;A \quad \dfrac{\Gamma;A\vdash\Lambda,\Omega;C}{\Gamma;A\vdash\Lambda,\Omega,B;C}\mathit{Aff}_d}{\Gamma;\vdash\Lambda,C,\Omega,B;A}\mathit{Coup}_\mu \quad \dfrac{\dfrac{\Gamma;A\vdash\Lambda,\Omega;C}{\Gamma;A\vdash\Lambda,C,\Omega;C}\mathit{Aff}_d}{\Gamma;A\vdash\Lambda,C,\Omega,B;C}\mathit{Aff}_d}{\Gamma;\vdash\Lambda,C,\Omega,B;C}\mathit{Coup}_T}{\Gamma;\vdash\Lambda,C,\Omega;B}\mathit{Chg}$$

$$\dfrac{\dfrac{\Gamma;\vdash\Lambda,D,\Omega,A,\Psi,B;D}{\Gamma;\vdash\Lambda,D,\Omega,A,\Psi;B}\mathit{Chg} \quad \Gamma;A\vdash\Lambda,D,\Omega,\Psi;C}{\Gamma;\vdash\Lambda,D,\Omega,C,\Psi;B}\mathit{Coup}_\mu \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma;\vdash\Lambda,D,\Omega,A,\Psi,B;D \quad \dfrac{\Gamma;A\vdash\Lambda,D,\Omega,\Psi;C}{\Gamma;A\vdash\Lambda,D,\Omega,\Psi,B;C}\mathit{Aff}_d}{\Gamma;\vdash\Lambda,D,\Omega,C,\Psi,B;D}\mathit{Coup}_\mu}{\Gamma;\vdash\Lambda,D,\Omega,C,\Psi;B}\mathit{Chg}$$

$$\dfrac{\dfrac{\Gamma,C;\vdash\Lambda,A,\Omega;D}{\Gamma;\vdash\Lambda,A,\Omega;C\to D}\to_d \quad \Gamma;A\vdash\Lambda,\Omega;B}{\Gamma;\vdash\Lambda,B,\Omega;C\to D}\mathit{Coup}_\mu \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma,C;\vdash\Lambda,A,\Omega;D \quad \dfrac{\Gamma;A\vdash\Lambda,\Omega;B}{\Gamma,C;A\vdash\Lambda,\Omega;B}\mathit{Aff}_g}{\Gamma,C;\vdash\Lambda,B,\Omega;D}\mathit{Coup}_\mu}{\Gamma;\vdash\Lambda,B,\Omega;C\to D}\to_d$$

$$\dfrac{\dfrac{\Gamma,D;D\vdash\Lambda,A,\Omega;C}{\Gamma,D;\vdash\Lambda,A,\Omega;C}\mathit{Déch} \quad \Gamma,D;A\vdash\Lambda,\Omega;B}{\Gamma,D;\vdash\Lambda,B,\Omega;C}\mathit{Coup}_\mu \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma,D;D\vdash\Lambda,A,\Omega;C \quad \Gamma,D;A\vdash\Lambda,\Omega;B}{\Gamma,D;D\vdash\Lambda,B,\Omega;C}\mathit{Coup}_\mu}{\Gamma,D;\vdash\Lambda,B,\Omega;C}\mathit{Déch}$$

- contrepartie logique de la diffusion d'une liste d'arguments à travers une liste d'arguments

$$\dfrac{\dfrac{\Gamma;\vdash \Lambda, A, \Omega; E \quad \Gamma; D \vdash \Lambda, A, \Omega; C}{\Gamma; E \to D \vdash \Lambda, A, \Omega; C}\to_g \qquad \Gamma; A \vdash \Lambda, \Omega; B}{\Gamma; E \to D \vdash \Lambda, B, \Omega; C}\,Coup_\mu$$

$$\overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma;\vdash \Lambda, A, \Omega; E \quad \Gamma; A \vdash \Lambda, \Omega; B}{\Gamma;\vdash \Lambda, B, \Omega; E}\,Coup_\mu \quad \dfrac{\Gamma; D \vdash \Lambda, A, \Omega; C \quad \Gamma; A \vdash \Lambda, \Omega; B}{\Gamma; D \vdash \Lambda, B, \Omega; C}\,Coup_\mu}{\Gamma; E \to D \vdash \Lambda, B, \Omega; C}\to_g$$

$$\dfrac{\dfrac{}{\Gamma; D \vdash \Lambda, A, \Omega; D}\,Ax \quad \Gamma; A \vdash \Lambda, \Omega; B}{\Gamma; D \vdash \Lambda, B, \Omega; D}\,Coup_\mu \qquad \overset{\text{LKT}}{\longrightarrow} \qquad \dfrac{}{\Gamma; D \vdash \Lambda, B, \Omega; D}\,Ax$$

Réduction de la règle Aff_g

- contrepartie logique des règles de calcul du décalage vers le haut à gauche à travers les termes faiblement normaux

$$\dfrac{\dfrac{\Gamma, \Delta, B;\vdash \Lambda; C}{\Gamma, \Delta;\vdash \Lambda; B \to C}\to_d}{\Gamma, A, \Delta;\vdash \Lambda; B \to C}\,\mathit{Aff}_g \qquad \overset{\text{LKT}}{\longrightarrow} \qquad \dfrac{\dfrac{\Gamma, \Delta, B;\vdash \Lambda; C}{\Gamma, A, \Delta, B;\vdash \Lambda; C}\,\mathit{Aff}_g}{\Gamma, A, \Delta;\vdash \Lambda; B \to C}\to_d$$

$$\dfrac{\dfrac{\Gamma, \Delta, B, \Phi; B \vdash \Lambda; C}{\Gamma, \Delta, B, \Phi;\vdash \Lambda; C}\,\mathit{Déch}}{\Gamma, A, \Delta, B, \Phi;\vdash \Lambda; C}\,\mathit{Aff}_g \qquad \overset{\text{LKT}}{\longrightarrow} \qquad \dfrac{\dfrac{\Gamma, \Delta, B, \Phi; B \vdash \Lambda; C}{\Gamma, A, \Delta, B, \Phi; B \vdash \Lambda; C}\,\mathit{Aff}_g}{\Gamma, A, \Delta, B, \Phi;\vdash \Lambda; C}\,\mathit{Déch}$$

et une règle similaire avec les places de A et B interverties.

$$\dfrac{\dfrac{\Gamma, \Delta;\vdash \Lambda, B, \Omega, C; B}{\Gamma, \Delta;\vdash \Lambda, B, \Omega; C}\,Chg}{\Gamma, A, \Delta;\vdash \Lambda, B, \Omega; C}\,\mathit{Aff}_g \qquad \overset{\text{LKT}}{\longrightarrow} \qquad \dfrac{\dfrac{\Gamma, \Delta;\vdash \Lambda, B, \Omega, C; B}{\Gamma, A, \Delta;\vdash \Lambda, B, \Omega, C; B}\,\mathit{Aff}_g}{\Gamma, A, \Delta;\vdash \Lambda, B, \Omega; C}\,Chg$$

- contrepartie logique des règles de calcul du décalage vers le haut à gauche à travers les listes d'arguments faiblement normales

$$\dfrac{\dfrac{}{\Gamma, \Delta; B \vdash \Lambda; B}\,Ax}{\Gamma, A, \Delta; B \vdash \Lambda; B}\,\mathit{Aff}_g \qquad \overset{\text{LKT}}{\longrightarrow} \qquad \dfrac{}{\Gamma, A, \Delta; B \vdash \Lambda; B}\,Ax$$

$$\dfrac{\dfrac{\Gamma, \Delta;\vdash \Lambda; B \quad \Gamma, \Delta; C \vdash \Lambda; D}{\Gamma, \Delta; B \to C \vdash \Lambda; D}\to_g}{\Gamma, A, \Delta; B \to C \vdash \Lambda; D}\,\mathit{Aff}_g \qquad \overset{\text{LKT}}{\longrightarrow} \qquad \dfrac{\dfrac{\Gamma, \Delta;\vdash \Lambda; B}{\Gamma, A, \Delta;\vdash \Lambda; B}\,\mathit{Aff}_g \quad \dfrac{\Gamma, \Delta; C \vdash \Lambda; D}{\Gamma, A, \Delta; C \vdash \Lambda; D}\,\mathit{Aff}_g}{\Gamma, A, \Delta; B \to C \vdash \Lambda; D}\to_g$$

Réduction de la règle Aff_d

- contrepartie logique des règles de calcul du décalage vers le haut à droite à travers les termes faiblement normaux

$$\dfrac{\dfrac{\Gamma, B; \vdash \Lambda, \Omega; C}{\Gamma; \vdash \Lambda, \Omega; B \to C}\,{\to_d}}{\Gamma; \vdash \Lambda, A, \Omega; B \to C}\,\mathit{Aff}_d \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma, B; \vdash \Lambda, \Omega; C}{\Gamma, B; \vdash \Lambda, A, \Omega; C}\,\mathit{Aff}_d}{\Gamma; \vdash \Lambda, A, \Omega; B \to C}\,{\to_d}$$

$$\dfrac{\dfrac{\Gamma, B, \Delta; B \vdash \Lambda, \Omega; C}{\Gamma, B, \Delta; \vdash \Lambda, \Omega; C}\,\mathit{Déch}}{\Gamma, B, \Delta; \vdash \Lambda, A, \Omega; C}\,\mathit{Aff}_d \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma, B, \Delta; B \vdash \Lambda, \Omega; C}{\Gamma, B, \Delta; B \vdash \Lambda, A, \Omega; C}\,\mathit{Aff}_d}{\Gamma, B, \Delta; \vdash \Lambda, A, \Omega; C}\,\mathit{Déch}$$

$$\dfrac{\dfrac{\Gamma; \vdash \Lambda, \Omega, B, \Psi, C; B}{\Gamma; \vdash \Lambda, \Omega, B, \Psi; C}\,\mathit{Chg}}{\Gamma; \vdash \Lambda, A, \Omega, B, \Psi; C}\,\mathit{Aff}_d \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma; \vdash \Lambda, \Omega, B, \Psi, C; B}{\Gamma; \vdash \Lambda, A, \Omega, B, \Psi, C; B}\,\mathit{Aff}_d}{\Gamma; \vdash \Lambda, A, \Omega, B, \Psi; C}\,\mathit{Chg}$$

et une règle similaire avec les places de A et B interverties.

- contrepartie logique des règles de calcul du décalage vers le haut à droite à travers les listes d'arguments faiblement normales

$$\dfrac{\dfrac{}{\Gamma; B \vdash \Lambda, \Omega; B}\,\mathit{Ax}}{\Gamma; B \vdash \Lambda, A, \Omega; B}\,\mathit{Aff}_d \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{}{\Gamma; B \vdash \Lambda, A, \Omega; B}\,\mathit{Ax}$$

$$\dfrac{\dfrac{\Gamma; \vdash \Lambda, \Omega; B \quad \Gamma; C \vdash \Lambda, \Omega; D}{\Gamma; B \to C \vdash \Lambda, \Omega; D}\,{\to_g}}{\Gamma; B \to C \vdash \Lambda, A, \Omega; D}\,\mathit{Aff}_d \quad \overset{\text{LKT}}{\longrightarrow} \quad \dfrac{\dfrac{\Gamma; \vdash \Lambda, \Omega; B}{\Gamma; \vdash \Lambda, A, \Omega; B}\,\mathit{Aff}_d \quad \dfrac{\Gamma; C \vdash \Lambda, \Omega; D}{\Gamma; C \vdash \Lambda, A, \Omega; D}\,\mathit{Aff}_d}{\Gamma; B \to C \vdash \Lambda, A, \Omega; D}\,{\to_g}$$

et a peu près les mêmes règles de réduction pour l'affaiblissement à droite.
Il y a aussi les règles de réduction de la μ-coupure.

4.5.4 L'assignement des preuves de LKT par des $\overline{\lambda}\mu$-expressions avec indices de de Bruijn

Règles de formation de contextes applicatifs

Liste vide d'arguments $$\frac{}{\Gamma; .:A \vdash \Lambda; (.\ [\,]) : A}\ Ax$$

Ajout d'un argument à une liste $$\frac{\Gamma; \vdash \Lambda; u : A \quad \Gamma; .:B \vdash \Lambda; (.\ l) : C}{\Gamma; .:A \to B \vdash \Lambda; (.\ [u :: l]) : C}\ \to_g$$

Concaténation de listes d'arguments $$\frac{\Gamma; .:C \vdash \Lambda; (.\ l) : A \quad \Gamma; .:A \vdash \Lambda; (.\ l') : B}{\Gamma; .:C \vdash \Lambda; (.\ (l \mathbin{@} l')) : B}\ Coup_T$$

Substitution dans une liste d'arguments $$\frac{\Gamma, \Delta; \vdash \Lambda; u : A \quad \Gamma, A, \Delta; .:C \vdash \Lambda; (.\ l) : B}{\Gamma, \Delta; .:C \vdash \Lambda; (.\ l[n := u]) : B}\ Coup_M$$

Diffusion dans une liste d'arguments $$\frac{\Gamma; .:D \vdash \Lambda, A, \Omega; (.\ l') : C \quad \Gamma; .:A \vdash \Lambda, \Omega; (.\ l) : B}{\Gamma; .:D \vdash \Lambda, B, \Omega; (.\ l')[[p]\chi := [p](\chi\ l)] : C}\ Coup_\mu$$

Rehaussement à gauche dans une liste d'arguments $$\frac{\Gamma, \Delta; C \vdash \Lambda; (.\ l) : B}{\Gamma, A, \Delta; C \vdash \Lambda; (.\ l{\uparrow_n}) : B}\ Aff_g$$

Rehaussement à droite dans une liste d'arguments $$\frac{\Gamma; C \vdash \Lambda, \Omega; u : B}{\Gamma; C \vdash \Lambda, A, \Omega; u{\uparrow^p} : B}\ Aff_d$$

Règles de formation de termes

Terme applicatif faiblement normal $$\frac{\Gamma, A, \Delta; .:A \vdash \Lambda; (.\ l) : B}{\Gamma, A, \Delta; \vdash \Lambda; (n\ l) : B}\ Déch$$

Abstraction $$\frac{\Gamma, A; \vdash \Lambda; u : B}{\Gamma; \vdash \Lambda; \lambda u : A \to B}\ \to_d$$

Changement de conclusion active $$\frac{\Gamma; \vdash \Lambda, A, \Omega, B; u : A}{\Gamma; \vdash \Lambda, A, \Omega; \mu[p]u : B}\ Chg$$

Terme applicatif non faiblement normal $$\frac{\Gamma; \vdash \Lambda; u : A \quad \Gamma; .:A \vdash \Lambda; (.\ l) : B}{\Gamma; \vdash \Lambda; (u\ l) : B}\ Coup_T$$

Substitution dans un terme $$\frac{\Gamma, \Delta; \vdash \Lambda; u : A \quad \Gamma, A, \Delta; \vdash \Lambda; v : B}{\Gamma, \Delta \vdash \Lambda; v[n := u] : B}\ Coup_M$$

Diffusion dans un terme $$\frac{\Gamma; \vdash \Lambda, A, \Omega; u : C \quad \Gamma; .:A \vdash \Lambda, \Omega; (.\ l) : B}{\Gamma; \vdash \Lambda, B, \Omega; u[[p]\chi := [p](\chi\ l)] : C}\ Coup_\mu$$

Rehaussement à gauche dans un terme $$\frac{\Gamma, \Delta; \vdash \Lambda; u : B}{\Gamma, A, \Delta; \vdash \Lambda; u{\uparrow_n} : B}\ Aff_g$$

Rehaussement à droite dans un terme $$\frac{\Gamma; \vdash \Lambda, \Omega; u : C}{\Gamma; \vdash \Lambda, A, \Omega; u{\uparrow^p} : C}\ Aff_d$$

où dans chaque règle mettant en jeu Δ, n est la longueur de Δ et dans chaque règle où apparaît p, c'est la longueur de Ω.

Un $\overline{\lambda}\mu$-terme u avec indices de de Bruijn tel qu'il existe des environnements Γ et Λ et une formule A de telle sorte que u soit l'assignement d'une preuve de $\Gamma; \vdash \Lambda; A$ est dit **simplement typable de type** A.

4.5.5 Terminaison forte

Nous montrons donc qu'une $\overline{\lambda}$-expression typable est fortement normalisable selon la réduction $\xrightarrow{1}$.

La preuve suit le même principe que la preuve du calcul sans indices de de Bruijn. La principale différence est la nécessité du lemme suivant :

Lemme 5 *Si e est FEN alors, pour tout p, $e{\uparrow_p}$ et $e{\uparrow^p}$ sont FEN*

qui permettra d'adapter en conséquence les preuves des lemmes 2, 3 et 4.

Voici une preuve du lemme :

PREUVE : Par induction sur l'arbre de E-réduction de e. On se contente du cas de $e{\uparrow_p}$

Soit e' tel que $e{\uparrow_p} \xrightarrow{1} e'$.

Si la réduction se passe dans e alors e' a la forme $e''{\uparrow_p}$, avec $e \xrightarrow{1} e''$. L'arbre de E-réduction de e'' étant plus petit que celui de e, par hypothèse d'induction, $e''{\uparrow_p}$ est FEN.

Si e est faiblement normal, $e{\uparrow_p}$ est un rédex et e' peut être le résultat de la réduction de ce rédex. Si e est λv alors e' est $\lambda(v{\uparrow_{p+1}})$. Comme $\lambda v \xrightarrow{h} v$, l'arbre de E-réduction de v est plus petit que celui de λv et, par hypothèse d'induction, $v{\uparrow_{p+1}}$ est FEN. Par le lemme 1, il s'ensuit que e' est aussi FEN.

Similairement pour les autres cas, de telle sorte que dans tous les cas possibles de réduction de e en e', ce dernier est FEN. On en déduit que e est FEN. ■

A partir des lemmes 1, 2, 3, 4 et 5, on déduit que toute expression typable est FEN et par conséquent fortement normalisable.

Calcul des séquents et stratégies gagnantes

Chapitre 5

LJQ, LKQ et E-dialogues de Lorenzen

Selon l'article de Felscher [17] dans le "Handbook of philosophical logic", c'est Lorenzen, qui, dans des communications de 1958 et 1959, publiées dans [38] et [39] (cf aussi [40]), proposa de fonder la prouvabilité sur des notions de jeux et de stratégies gagnantes pour ces jeux.

Les jeux qui sont considérés par Lorenzen sont à propos d'une formule. Deux joueurs sont en présence. Le premier, appellons-le le proposant, tente de prouver la validité de la formule, l'autre, appelons-le l'opposant, tente de réfuter la formule. De la formule, chaque connecteur et quantificateur de la formule peut être successivement attaqué par un joueur et défendu par l'autre. A chaque connecteur et quantificateur correspond un certain type d'attaques possibles et un certain type de réponses (= défenses) en conséquence.

A partir de ces principes généraux, il s'agit d'établir des règles de jeux, gérant les possibilités d'attaques et les moyens de défense d'une formule, de telle sorte que l'existence d'une stratégie gagnante pour le proposant dans un jeu à propos d'une certaine formule A, soit équivalente à la prouvabilité de A.

Toujours selon Felscher, plusieurs années se sont écoulées avant de trouver des principes de jeux tels qu'il y ait effectivement cette équivalence entre prouvabilité et existence d'une stratégie gagnante pour le proposant. Les E-dialogues, définis par Lorenzen (cf Felscher [17, 18] pour un historique des E-dialogues), conviennent pour cet objectif. Et c'est à Felscher qu'est due la démonstration de l'équivalence entre l'existence d'une stratégie gagnante pour le proposant dans un E-dialogue sur une formule A et la prouvabilité de A dans le calcul des séquents LJ. Sa preuve, valable pour le cas de la logique intuitionniste, utilise plusieurs notions intermédiaires permettant une transformation progressive de stratégies gagnantes en preuves et vice-versa. Ici, nous exhibons directement une variante complète de LJ telle que les preuves d'une formule A soient en bijection avec les stratégies gagnantes pour un E-dialogue à propos de A.

Présumable arbitraire, cette variante, dans le cas du fragment restreint à $\wedge$ et $\rightarrow$ de la logique propositionnelle, apparaît curieusement comme la restriction d'un calcul classique LKQ que Danos, Joinet et Schellinx [14] avaient mis en avant pour ses bonnes propriétés vis à vis de la logique linéaire (de telle sorte que de ces deux calculs, LKT et LKQ, définis par Danos *et al* dans le but d'analyser le comportement calculatoire de la logique au travers du comportement calculatoire de la logique linéaire suscitent curieusement, et l'un, et l'autre, chacun dans un cadre différent, un intérêt nouveau). Appelons LJQ cette restriction de LKQ à la logique intuitionniste.

Le cas de formules avec disjonction pose un problème supplémentaire : on n'a plus bijection avec l'extension "naturelle" de LJQ au $\vee$. En revanche, si l'on considère une version de la logique intuitionniste avec plusieurs formules à droite du symbole $\vdash$, là on garde une bijection entre preuves et stratégies gagnantes pour E-dialogues.

Nous croyons que cette manière de mettre en bijection preuves de certains calculs et stratégies gagnantes pour certains types de jeux à deux joueurs se disputant la validité d'une formule peut se faire de manière quasi-systématique, quelque soit le calcul considéré, qu'il soit intuitionniste ou classique. Nous pourrions le montrer, par exemple, pour LJT dont nous conjecturons les preuves en correspondance avec les stratégies innocentes de Hyland et Ong [33].

Une telle bijection est, par exemple, décrite dans le chapitre suivant. Dans ce prochain chapitre, c'est un calcul, inspirée de travaux de Novikoff, qui est introduit, justement pour la simplicité de son interprétabilité en termes de jeux. Coquand, qui est l'origine de l'interprétation de ce "calcul de Novikoff" en termes de jeux y a défini une manière de faire interagir des preuves vues comme stratégies gagnantes. Nous montrons un lien entre ce procédé d'interaction entre preuves et lélimination des coupures.

La question demeure cependant de déterminer entre plusieurs variantes de jeux, celles qui répondent à une intuition acceptable.

5.1 E-dialogues de Felscher

On considère les formules construites à partir de $\rightarrow$, $\wedge$, $\vee$ et les variables propositionnelles.

A chaque formule, on associe un ensemble d'attaques possibles de cette formule ainsi que, pour chacune de ces attaques, un ensemble de réponses à cette attaque.

Pour noter les attaques, on utilisera, entre autres, les symboles suivants, dits **symboles d'attaque** : $\wedge_1$, $\wedge_2$ et $\vee$.

On note les attaques sous la forme $([, s)$ où s est soit un symbole d'attaque, soit une formule. Les réponses, quant à elles se notent sous la forme $(], A)$ où A est une formule.

Les attaques de la forme $([,A)$ et les réponses $(],A)$ sont appelées **affirmations** de A. Ce sont les affirmations qui sont susceptibles d'être attaquées. Ainsi, il y a deux types d'attaques, celles pouvant être attaquées, de la forme $([,A)$, et les autres, de la forme $([,s)$ avec s symbole d'attaque, qui ne peuvent pas être attaquées. En revanche, toutes les réponses affirmant des formules non atomiques sont attaquables.

La caractérisation des attaques est la suivante :

- $([,B)$ est une attaque de $B \to A$
- $([,\wedge_1)$ et $([,\wedge_2)$ sont des attaques de $A \wedge B$
- $([,\vee)$ est une attaque de $A \vee B$

A chacune de ces attaques, les **réponses** possibles sont les suivantes :

- $(],A)$ est une réponse à l'attaque de $B \to A$ par B
- $(],A)$ est une réponse à l'attaque de $A \wedge B$ par $\wedge_1$
- $(],B)$ est une réponse à l'attaque de $A \wedge B$ par $\wedge_2$
- $(],A)$ et $(],B)$ sont chacun une réponse à l'attaque de $A \vee B$ par $\vee$

Pour mémoriser les affirmations auxquelles les attaques font référence, et pour mémoriser les attaques auxquelles les réponses font référence, on considère des attaques et des réponses indexées par des entiers.

Un **coup d'attaque** est la paire d'un entier et d'une attaque. Un **coup réponse** est la paire d'un entier et d'une réponse.

Soit $d_0, d_1, ...$ une suite de coups.

On dit qu'un coup d'attaque $(n,[,s)$ est **justifié** si d_n affirme A (c'est-à-dire que d_n a la forme $(m,],A)$ ou bien $(m,[,A)$) et s attaque A où s est une attaque.

On dit qu'un coup réponse $(n,],A)$ est **justifié** si le coup d_n, ayant la forme $(m,[,s)$, est justifié avec s attaquant B et i A est une réponse à l'attaque de B par s.

On dit qu'une réponse $(n,],A)$ est **une reprise** s'il existe $l < n$ et de parité distincte de celle de n, tel que d_l soit une affirmation de A (c'est-à-dire tel que d_l ait la forme $(],l',A)$ ou $([,l',A)$).

On dit que la suite $d_1, d_2, ...$ est **potentiellement bien parenthésée** si la suite sous-jacente de "$(n,[)$" et de "$(n,])$" définit une expression complétable en une expression bien parenthésée (chaque coup réponse de la forme $(n,],A)$ devant répondre à une attaque jouée au coup n.

Une **partie** d sur A est une suite finie ou infinie de coups telle que

- le premier coup a la forme $d_0 = (],0,A)$
- d est potentiellement bien parenthésé à partir de d_1
- chaque coup d_n, pour $n > 0$, est justifié
- si de plus $d_n = (m,],B)$ ou $d_n = (m,[,B)$ avec B atomique et n pair alors d_n est une reprise

Un coup d est dit **permis** après une suite de coups $d_0, d_1, ..., d_n$ si la suite de coups $d_0, d_1, ..., d_n, d$ constitue une partie.

Les coups de numéro pair sont appelés coups du proposant (en abrégé **P-coups**) et les coups de numéro impair sont appelés coups de l'opposant (en abrégé **O-coups**).

On dit qu'une partie est **gagnée** pour le proposant et **perdue** pour l'opposant si elle est finie et que l'opposant n'a plus de coups permis. Dans le cas contraire, elle est gagnée par l'opposant et perdue pour le proposant.

Une **P-stratégie** pour A est la donnée d'un arbre représenté comme un ensemble $D(\phi)$ de suites finies de O-coups et d'une fonction ϕ de $D(\phi)$ vers l'ensemble des P-coups telle que

- la suite vide ϵ appartient à $D(\phi)$
- si $d_1 d_3 ... d_{2n+1} \in D(\phi)$ le O-coup d_{2n+3} est permis après

$$d_0, d_1, \phi(d_1), d_3, \phi(d_1 d_3), ..., d_{2n+1}, \phi(d_1 d_3 ... d_{2n+1}),$$

  si et seulement si $d_1 d_3 ... d_{2n+3} \in D(\phi)$
- $\phi(d_1 d_3 ... d_{2n-1})$ est permis après

$$d_0, d_1, \phi(d_1), d_3, \phi(d_1 d_3), ..., d_{2n-1}$$

Une P-stratégie ϕ est **gagnante** si $D(\phi)$ est bien fondé, autrement dit si, quelques soient les coups de l'opposant, au bout d'un temps fini, la partie est perdue pour lui. On admet qu'un principe d'induction bien-fondée est associé à la bonne fondaison de $D(\phi)$.

Remarque : La notion de partie est indicative du genre de jeu qui se déroule entre preuves. Dans la suite, nous considérons essentiellement un certain type de parties appelées E-dialogues. Nous évoquerons aussi le cas des D-dialogues, considérés par exemple dans Felscher [17, 18].

Un **E-dialogue** est une partie pour laquelle les O-coups doivent systématiquememt répondre au P-coup précédent. Autrement dit, si n est pair, d_n a la forme $(n-1, [, s)$ ou bien $(n-1,], A)$. Cette définition correspond précisément aux E-dialogues définis par Lorenzen et décrits par Felscher dans [17].

On appelle **PE-stratégie** sur A toute P-stratégie pour un E-dialogue sur A.

Exemple :

Considérons la formule $(A \to B) \to (B \to C) \to (A \to C)$.
Une partie sur cette formule a cette forme :

Coup initial : P affirme $(A \to B) \to (B \to C) \to (A \to C)$
$d_0 = (0,], (A \to B) \to (B \to C) \to (A \to C))$

O ne peut qu'attaquer l'implication
$d_1 = (0, [, A \to B)$

P répond/affirme $(B \to C) \to (A \to C)$
$d_2 = (1,], (B \to C) \to (A \to C))$

O ne peut qu'attaquer $(B \to C) \to (A \to C)$
$d_3 = (2, [, B \to C)$

P répond/affirme $A \to C$
$d_4 = (3,], A \to C)$

O ne peut qu'attaquer $A \to C$
$d_5 = (4, [, A)$

P attaque $A \to B$ (c'est une reprise de A)
$d_6 = (1, [, A)$

O ne peut que répondre B
$d_7 = (6,], B)$

P attaque alors $B \to C$ (c'est une reprise de B)
$d_8 = (3, [, B)$

O ne peut que répondre C
$d_9 = (8,], C)$

P peut alors à son tour répondre C (c'est une reprise de C)
$d_10 = (5,], C)$

Et O ne peut plus jouer.

L'opposant n'a jamais eu qu'un seul choix à chaque étape si bien que la partie ci-dessus correspond directement à une PE-stratégie gagnante. Un autre exemple de partie apparaît en section 5.2.3.

5.2 Correspondance entre preuves de LJQ et PE-stratégies gagnantes

5.2.1 Le calcul des séquents LJQ

On décrit LJQ qui est un calcul des séquents sans coupure. Ce calcul est un fragment propre de LJ. On adopte une définition de ce calcul pour laquelle les parties gauches des séquents sont des ensemble de formules nommées et pour laquelle les règles structurelles sont intégrées aux autres règles. On se limite au fragment propositionnel avec $\rightarrow$, $\wedge$. et $\vee$.

$$\frac{}{\Gamma, A \vdash A}\ Ax \text{ si } A \text{ est atomique}$$

$$\frac{\Gamma, A \rightarrow B \vdash A \quad \Gamma, A \rightarrow B, B \vdash C}{\Gamma, A \rightarrow B \vdash C} \rightarrow_g \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \rightarrow_d$$

$$\frac{\Gamma, A \wedge B, A \vdash C}{\Gamma, A \wedge B \vdash C} \wedge_g^1 \qquad \frac{\Gamma, A \wedge B, B \vdash C}{\Gamma, A \wedge B \vdash C} \wedge_g^2 \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge_d$$

$$\frac{\Gamma, A \vee B, A \vdash C \quad \Gamma, A \vee B, B \vdash C}{\Gamma, A \vee B \vdash C} \vee_g \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \vee_d^1 \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \vee_d^2$$

Le calcul LJQ se différencie de LJ par cette contrainte :

> Sur le côté gauche de la règle $\rightarrow_g$, seules les règles Ax, $\wedge_d$, $\vee_d^1$, $\vee_d^2$ et $\rightarrow_d$ sont autorisées (i.e. l'axiome et les règles d'introduction droite)

LJQ est complet pour la logique minimale. Le nom de LJQ vient du fait que ce calcul apparaît comme le fragment intuitionniste du calcul des séquent LKQ pour la logique classique, défini par Danos *et al* dans [14].

5.2.2 La correspondance pour les formules sans disjonction

On se restreint ici à LJQ sans disjonction.

Dans la table suivante, on associe à chaque règle de LJQ une attaque ou une réponse du proposant. On mentionne aussi les attaques ou réponses de l'opposant justifiées par le coup du proposant.

Règle	P-coup	O-coup justifié
$\dfrac{}{\Gamma, A \vdash A}\ Ax$	$(], A)$	Aucun O-coup justifié
$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \to_d$	$(], A \to B)$	$([, A)$
$\dfrac{\Gamma, A \to B \vdash A \quad \Gamma, A \to B, B \vdash C}{\Gamma, A \to B \vdash C} \to_g$	$([, A)$	$(], B)$ ou $([, s)$
$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge_d$	$(], A \wedge B)$	$([, \wedge_1)$ ou $([, \wedge_2)$
$\dfrac{\Gamma, A \wedge B, A \vdash C}{\Gamma, A \wedge B \vdash C} \wedge_g^1$	$([, \wedge_1)$	$(], A)$
$\dfrac{\Gamma, A \wedge B, B \vdash C}{\Gamma, A \wedge B \vdash C} \wedge_g^2$	$([, \wedge_2)$	$(], B)$

Sur la troisième ligne, s attaque A.

Soit F une formule sans disjonction. Soit p une preuve de A dans LJQ.

Cette preuve a une structure d'arbre et nous associons à cet arbre un ensemble de suite de O-coups constituant le domaine $D(\phi)$ d'une certaine P-stratégie. Notons que pour la règle $\to_g$, on ne considére pas directement la sous-preuve de gauche, mais uniquement les sous-preuves directes de cette sous-preuve de gauche.

Soit q une sous-preuve de p et $\vec{d}$ une suite d'attaques et de réponses. On définit une relation $\vec{d} : q$ par les cas suivants :

- $\epsilon : p$ (ϵ est la suite vide)

- si $\vec{d} : \dfrac{\begin{matrix} \vdots\, q_1 & \vdots\, q_2 \\ \Gamma \vdash A & \Gamma \vdash B \end{matrix}}{\Gamma \vdash A \wedge B} \wedge_d$ alors $\vec{d}([, \wedge_1) : q_1$ et $\vec{d}([, \wedge_2) : q_2$

- si $\vec{d} : \dfrac{\begin{matrix} \vdots\, q \\ \Gamma, A \vdash B \end{matrix}}{\Gamma \vdash A \to B} \to_d$ alors $\vec{d}([, A) : q$

- si $\vec{d} : \dfrac{\begin{matrix} \vdots\, q \\ \Gamma, A \wedge B, A \vdash C \end{matrix}}{\Gamma, A \wedge B \vdash C} \wedge_g^1$ alors $\vec{d}(], A) : q$

- si $\vec{d} : \dfrac{\begin{matrix} \vdots\, q \\ \Gamma, A \wedge B, B \vdash C \end{matrix}}{\Gamma, A \wedge B \vdash C} \wedge_g^2$ alors $\vec{d}(], B) : q$

- si $\vec{d} : \dfrac{\dfrac{}{\Gamma, A \to B, A \vdash A}\ Ax \qquad \begin{matrix} \vdots\, q \\ \Gamma, A \to B, B, A \vdash C \end{matrix}}{\Gamma, A \to B, A \vdash C} \to_g$ alors $\vec{d}(], B) : q$

- si $\vec{d}$: $\dfrac{\dfrac{\overset{\vdots\, q_1}{\Gamma, A \wedge B \to C \vdash A} \quad \overset{\vdots\, q_2}{\Gamma, A \wedge B \to C \vdash B}}{\Gamma, A \wedge B \to C \vdash A \wedge B} \wedge_d \quad \overset{\vdots\, q}{\Gamma, A \wedge B \to C, C \vdash D}}{\Gamma, A \wedge B \to C \vdash D} \to_g$

  alors $\vec{d}([, \wedge_1) : q_1$, $\vec{d}([, \wedge_2) : q_2$ et $\vec{d}(], C) : q_3$

- si $\vec{d}$: $\dfrac{\dfrac{\overset{\vdots\, q_1}{\Gamma, (A \to B) \to C, A \vdash B}}{\Gamma, (A \to B) \to C \vdash A \to B} \to_d \quad \overset{\vdots\, q_2}{\Gamma, (A \to B) \to C, C \vdash D}}{\Gamma, (A \to B) \to C \vdash D} \to_g$ alors $\vec{d}([, A) : q_1$ et $\vec{d}(], C) : q_2$

Par construction, si $\vec{d} : q_1$ et $\vec{d} : q_2$, alors q_1 et q_2 sont les mêmes preuves. Ainsi la relation $\vec{d} : q$ définit une fonction partielle de l'ensemble des suites d'occurrences vers les sous-preuves de p. On pose $D(\phi)$ le domaine de définition de cette fonction.

Pour $\vec{d}$ tel que $\vec{d} : q$ et q prouve $\Gamma \vdash A$, on définit $\sigma_{\vec{d}}$ et $a_{\vec{d}}$.
$\sigma_{\vec{d}}$ est une fonction des noms utilisés dans Γ vers l'ensemble des entiers naturels impairs.
$a_{\vec{d}}$ est un entier impair ou bien 0.
On note $|\vec{d}|$ la longueur de la suite $\vec{d}$.

- $\sigma_\epsilon = \emptyset$
  $a_\epsilon = 0$
- $\sigma_{\vec{d}([, \wedge_1)} = \sigma_{\vec{d}([, \wedge_2)} = \sigma_{\vec{d}}$
  $a_{\vec{d}([, \wedge_1)} = a_{\vec{d}([, \wedge_2)} = a_{\vec{d}([, A)} = 2|\vec{d}| + 1$
- $\sigma_{\vec{d}([, A)} = \sigma_{\vec{d}(], A)} = \sigma_{\vec{d}} \cup \{x \mapsto 2|\vec{d}| + 1\}$ où x est, chaque fois, le nom de A dans $\Gamma, A \vdash B$
  $a_{\vec{d}(], A)} = a_{\vec{d}}$

Sur $D(\phi)$, on définit ϕ :

- si $\vec{d}$: $\dfrac{}{\Gamma, A \vdash A} Ax$ alors $\phi(\vec{d}) = (a_{\vec{d}},], A)$
- si $\vec{d}$: $\dfrac{\overset{\vdots}{\Gamma \vdash A} \quad \overset{\vdots}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B} \wedge_d$ alors $\phi(\vec{d}) = (a_{\vec{d}},], A \wedge B)$
- si $\vec{d}$: $\dfrac{\overset{\vdots}{\Gamma, A \vdash B}}{\Gamma \vdash A \to B} \to_d$ alors $\phi(\vec{d}) = (a_{\vec{d}},], A \to B)$
- si $\vec{d}$: $\dfrac{\overset{\vdots}{\Gamma, A \wedge B, A \vdash C}}{\Gamma, A \wedge B \vdash C} \wedge_g^1$ alors $\phi(\vec{d}) = (\sigma_{\vec{d}}(y), [, \wedge_1)$ où y est le nom de $A \wedge B$
- si $\vec{d}$: $\dfrac{\overset{\vdots}{\Gamma, A \wedge B, B \vdash C}}{\Gamma, A \wedge B \vdash C} \wedge_g^2$ alors $\phi(\vec{d}) = (\sigma_{\vec{d}}(y), [, \wedge_2)$ où y est le nom de $A \wedge B$

- si $\vec{d} : \dfrac{\Gamma, A \to B \vdash A \quad \Gamma, A \to B, B \vdash C}{\Gamma, A \to B \vdash C} \to_g$ alors $\phi(\vec{d}) = (\sigma_{\vec{d}}(y), [, A)$ où y est le nom de $A \to B$

Proposition 4 *$(D(\phi), \phi)$ est une PE-stratégie gagnante pour F.*

PREUVE : On vérifie que $\epsilon \in D(\phi)$. Posant, pour $d_1 d_3 ... d_{2n+1} \in D(\phi)$, $d_{2n+2} = \phi(d_1 d_3 ... d_{2n+1})$, on vérifie par induction sur n que $d_0, d_1, ..., d_n$ est une partie, que si n est pair d_{n+1} est permis si et seulement si $d_1 d_3 ... d_{n+1} \in \vec{d}(\phi)$ et que si n est impair $\phi(d_1 d_3 ... d_n)$ est permis.

Le fait que la preuve p soit bien fondée entraine que la stratégie ainsi constituée est gagnante. ■

Regardons la réciproque.
Soit $(D(\phi), \phi)$ une PE-stratégie gagnante pour F.

A toute suite $\vec{d} \in D(\phi)$, on associe un séquent $\Gamma_{\vec{d}} \vdash A_{\vec{d}}$.

- $\Gamma_\epsilon = \emptyset \qquad A_\epsilon = F$
- $\Gamma_{\vec{d}(],A)} = \Gamma_{\vec{d}}, A \qquad A_{\vec{d}(],A)} = A_{\vec{d}}$
- $\Gamma_{\vec{d}([,\wedge_1)} = \Gamma_{\vec{d}} \qquad A_{\vec{d}}$ doit avoir la forme $A \wedge B$ et on pose $A_{\vec{d}([,\wedge_1)} = A$
- $\Gamma_{\vec{d}([,\wedge_2)} = \Gamma_{\vec{d}} \qquad A_{\vec{d}}$ doit avoir la forme $A \wedge B$ et on pose $A_{\vec{d}([,\wedge_2)} = B$
- $\Gamma_{\vec{d}([,A)} = \Gamma_{\vec{d}}, A \qquad A_{\vec{d}}$ doit avoir la forme $A \to B$ et on pose $A_{\vec{d}([,A)} = B$

Pour les noms associés aux formules des $\Gamma_{\vec{d}}$, on choisit des entiers. A chaque nouvelle formule, on associe l'entier $|\vec{d}| + 1$.

Remarque : De par les règles de formation de $D(\phi)$, une attaque par l'opposant d'une formule A impose que cette formule vient juste d'être affirmée. La formule $A_{\vec{d}}$ a donc la forme souhaitée dans chacun des cas d'attaques de l'opposant.

A tout $\vec{d} \in D(\phi)$ on associe une preuve $p_{\vec{d}}$ de $\Gamma_{\vec{d}} \vdash A_{\vec{d}}$.

- si $\phi(\vec{d}) = (m, [, \wedge_1)$ est une attaque de $A \wedge B$ alors $p_{\vec{d}} = \dfrac{\vdots\, p_{\vec{d}(],A)} \atop \Gamma, A \wedge B, A \vdash C}{\Gamma, A \wedge B \vdash C} \wedge_g^1$

- si $\phi(\vec{d}) = (m, [, \wedge_2)$ est une attaque de $A \wedge B$ alors $p_{\vec{d}} = \dfrac{\vdots\, p_{\vec{d}(],B)} \atop \Gamma, A \wedge B, B \vdash C}{\Gamma, A \wedge B \vdash C} \wedge_g^2$

- si $\phi(\vec{d}) = (m, [, A \wedge B)$ est une attaque de $A \wedge B \to C$ alors

$$p_{\vec{d}} = \dfrac{\dfrac{{\vdots\, p_{\vec{d}([,\wedge_1)} \atop \Gamma, A \wedge B \to C \vdash A} \quad {\vdots\, p_{\vec{d}([,\wedge_2)} \atop \Gamma, A \wedge B \to C \vdash B}}{\Gamma, A \wedge B \to C \vdash A \wedge B} \wedge_d \quad {\vdots\, p_{\vec{d}(],C)} \atop \Gamma, A \wedge B \to C, C \vdash D}}{\Gamma, A \wedge B \to C \vdash D} \to_g$$

- si $\phi(\vec{d}) = (m, [, A \to B)$ est une attaque de $(A \to B) \to C$ alors

$$p_{\vec{d}} = \dfrac{\dfrac{\dfrac{\vdots\, p_{\vec{d}([,A)}}{\Gamma, (A \to B) \to C, A \vdash B}}{\Gamma, (A \to B) \to C \vdash A \to B} \to_d \quad \dfrac{\vdots\, p_{\vec{d}(],C)}}{\Gamma, (A \to B) \to C, C \vdash D}}{\Gamma, (A \to B) \to C \vdash D} \to_g$$

- si $\phi(\vec{d}) = (m, [, A)$ est une attaque de $A \to B$ avec A atomique alors

$$p_{\vec{d}} = \dfrac{\dfrac{}{\Gamma, A \to B, A \vdash A}\, Ax \quad \dfrac{\vdots\, p_{\vec{d}(],B)}}{\Gamma, A \to B, B, A \vdash C}}{\Gamma, A \to B, A \vdash C} \to_g$$

- si $\phi(\vec{d}) = (m,], A)$ avec A atomique alors $p_{\vec{d}} = \dfrac{}{\Gamma, A \vdash A}\, Ax$

- si $\phi(\vec{d}) = (m,], A \wedge B)$ alors $p_{\vec{d}} = \dfrac{\dfrac{\vdots\, p_{\vec{d}([,\wedge_1)}}{\Gamma \vdash A} \quad \dfrac{\vdots\, p_{\vec{d}([,\wedge_2)}}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B} \wedge_d$

- si $\phi(\vec{d}) = (m,], A \to B)$ alors $p_{\vec{d}} = \dfrac{\dfrac{\vdots\, p_{\vec{d}([,A)}}{\Gamma, A \vdash B}}{\Gamma \vdash A \to B} \to_d$

Proposition 5 *p_ϵ est une preuve de $\vdash F$ dans LJQ.*

PREUVE : Par induction bien fondée sur $\vec{d}$, montrant que $p_{\vec{d}}$ est une preuve de $\Gamma_{\vec{d}} \vdash F_{\vec{d}}$. L'induction est bien fondée du fait que la stratégie est gagnante. ∎

Théorème 1 *Si p_ϵ est la preuve de $\vdash F$ associée à la PE-stratégie gagnante $(D(\phi), \phi)$ pour F et si $(D'(\phi'), \phi')$ est la PE-stratégie gagnante pour F asociée à p_ϵ, alors $D(\phi) = D'(\phi')$ et $\phi = \phi'$. De plus, la structure d'arbre de p_ϵ est isomorphe à la structure d'arbre de $D(\phi)$.*

PREUVE : Soit $\vec{d}$ une suite de coups. On montre par induction sur la longueur de $\vec{d}$, et en envisageant les coups permis à chaque étape, que, *relativement* à p_ϵ, on a $\vec{d} : q$ si et seulement si $\vec{d} \in D(\phi)$ et en ce cas, on a $q = p_{\vec{d}}$. On en déduit que $D(\phi) = D'(\phi')$. Mais la règle de tête de $p_{\vec{d}}$ ne dépend que de $\phi(\vec{d})$ et en retour, $\phi'(\vec{d})$ ne dépend que de la règle de tête de $p_{\vec{d}}$. On en déduit que $\phi = \phi'$. L'isomorphisme entre les structures d'arbres sous-jacente découle de la définition de la bijection. ∎

Remarque : A chaque étape, si la règle considérée prouve un séquent $\Gamma \vdash A$, alors les propriétés suivantes sont satisfaites :

- la seule P-réponse justifiée a la forme $(m,], A)$
- les seules P-attaques justifiées ont la forme $(m, [, s)$ où s attaque une formule C in Γ

5.2.3 Le cas des formules avec disjonction

La prise en compte de la disjonction ne permet pas d'établir une correspondance entre E-dialogues et preuves de LJQ aussi "naturelle" que précédemment.

Le principal problème est celui-là : pour un E-dialogue, lorsqu'un joueur affirme une proposition de la forme $A \vee B$, il peut se donner la possibilité d'attendre le dernier moment avant de répondre $(], A)$ ou $(], B)$

à une attaque $([, \vee)$ de $A \vee B$ par l'adversaire. Ce n'est pas ce qui se passe dans LJQ, où, poursuivant le principe qu'une règle d'introduction droite correspond à une réponse/affirmation, lorsque $A \vee B$ est affirmé, la preuve stipule à l'avance quelle sera la réponse à l'attaque de $A \vee B$ par l'adversaire.

L'exemple typique est le suivant, tiré de Felscher [17].

C'est un jeu sur $A \vee B \rightarrow B \vee A$.

Coup initial : P répond/affirme $A \vee B \rightarrow B \vee A$
$d_0 = (0,], A \vee B \rightarrow B \vee A)$

O ne peut qu'attaquer l'implication
$d_1 = (0, [, A \vee B)$

Là, deux suites sont possibles :

P attaque $A \vee B$ $d_2 = (1, [, \vee)$	P répond/affirme $B \vee A$ $d_2 = (1,], B \vee A)$
O répond A (resp B) $d_3 = (2,], A)$ (resp $d_3 = (2,], B)$)	O attaque $B \vee A$ $d_3 = (2, [, \vee)$
P répond/affirme $B \vee A$ $d_4 = (1,], B \vee A)$	P attaque $A \vee B$ $d_4 = (1, [, \vee)$
O attaque $B \vee A$ $d_5 = (4, [, \vee)$	O répond A (resp B) $d_5 = (4,], A)$ (resp $d_5 = (4,], B)$)
P reprend/affirme A (resp B) $d_6 = (5,], A)$ (resp $d_6 = (5,], B)$)	P reprend/affirme A (resp B) $d_6 = (3,], A)$ (resp $d_6 = (3,], B)$)

La partie de droite définit une stratégie gagnante qui a une contrepartie sur le plan des preuves. Mais pas la seconde partie.

Face à ce problème, une solution est de considérer une version de LJQ avec plusieurs formules à droite du $\vdash$.

Le calcul LJQ* associé aux E-dialogues en présence de disjonction

On appelle LJ* la version suivante de LJ (la règle d'introduction du $\vee$ est multiplicative) :

$$\frac{}{\Gamma, A \vdash \Delta, A}\ Ax \text{ si } A \text{ est atomique}$$

$$\frac{\Gamma, A \rightarrow B \vdash A \quad \Gamma, A \rightarrow B, B \vdash \Delta, C}{\Gamma, A \rightarrow B \vdash \Delta, C} \rightarrow_g \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash \Delta, A \rightarrow B} \rightarrow_d$$

$$\frac{\Gamma, A \wedge B, A \vdash \Delta, C}{\Gamma, A \wedge B \vdash \Delta, C} \wedge_g^1 \qquad \frac{\Gamma, A \wedge B, B \vdash \Delta, C}{\Gamma, A \wedge B \vdash \Delta, C} \wedge_g^2 \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash \Delta, A \wedge B} \wedge_d$$

$$\frac{\Gamma, A \vee B, A \vdash \Delta, C \quad \Gamma, A \vee B, B \vdash \Delta, C}{\Gamma, A \vee B \vdash \Delta, C} \vee_g \qquad \frac{\Gamma \vdash A, B}{\Gamma \vdash \Delta, A \vee B} \vee_d$$

On définit LJQ* à partir de LJ* en posant la même restriction que pour LJQ, à savoir que sur le côté gauche de la règle $\rightarrow_g$, seules les règles Ax, $\wedge_d$, $\vee_d$ et $\rightarrow_d$ sont autorisées.

La correspondance

La correspondance entre règles et coups est alors la suivante :

Règle	P-coup	O-coup justifié
$\dfrac{}{\Gamma, A \vdash \Delta, A}\ Ax$	$(], A)$	Aucun O-coup justifié
$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash \Delta, A \to B} \to_d$	$(], A \to B)$	$([, A)$
$\dfrac{\Gamma, A \to B \vdash A \quad \Gamma, A \to B, B \vdash \Delta, C}{\Gamma, A \to B \vdash \Delta, C} \to_g$	$([, A)$	$(], B)$ ou $([, s)$
$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash \Delta, A \wedge B} \wedge_d$	$(], A \wedge B)$	$([, \wedge_1)$ ou $([, \wedge_2)$
$\dfrac{\Gamma, A \wedge B, A \vdash \Delta, C}{\Gamma, A \wedge B \vdash \Delta, C} \wedge_g^1$	$([, \wedge_1)$	$(], A)$
$\dfrac{\Gamma, A \wedge B, B \vdash \Delta, C}{\Gamma, A \wedge B \vdash \Delta, C} \wedge_g^2$	$([, \wedge_2)$	$(], B)$
$\dfrac{\Gamma \vdash A, B}{\Gamma \vdash \Delta, A \vee B} \vee_d$	$(], A \vee B)$	$([, \vee)$
$\dfrac{\Gamma, A \vee B, A \vdash \Delta, C \quad \Gamma, A \vee B, B \vdash \Delta, C}{\Gamma, A \vee B \vdash \Delta, C} \vee_g$	$([, \vee)$	$(], A)$ ou $(], B)$

et avec le même canevas de preuve que précédemment, en vérifiant règle par règle la correspondance entre les coups permis et la forme des prémisses, on prouve la correspondance entre E-dialogues et preuves de LJQ*.

La manière d'autoriser plusieurs formules à droite du $\vdash$ est ici un peu artificielle, cependant il ne semble pas déraisonnable de penser qu'en considérant carrément les versions traditionnelles de la logique intuitionniste avec plusieurs formules à droite du $\vdash$, on obtiendrait une nouvelle définition de E-dialogues moins coercitive au niveau du parenthésage que l'actuelle version des E-dialogues.

5.3 Le calcul LKQ* et les jeux non bien parenthésés

Des résultats similaires existent avec la logique classique. Cette fois, la correspondance est entre les preuves de LKQ* (un calcul étendant LJQ* à la logique classique) et les P-stratégies gagnantes pour un E-dialogue classique, ce dernier étant un E-dialogue, au sens précédent, dont on a supprimé la contrainte de bon parenthésage. Il est d'ailleurs remarquable que la notion de jeux pour laquelle on a une correspondance avec la logique classique soit "plus simple" que celle pour la logique intuitionniste.

5.3.1 Les E-dialogues pour la logique classique

Un E-dialogue pour la logique classique est un E-dialogue pour la logique intuitionniste sans la règle de potentiel bon-parenthèsage.

5.3.2 Le calcul des séquents LKQ*

Nous donnons la définition de LKQ*. Ce calcul étend LJQ* au cas classique. Il diffère de LKQ considéré par Joinet [35] ou Schellinx [51] par la présence d'une règle d'introduction "multiplicative" pour le $\vee$. La définition de LKQ passe par la définition de LK* qui est une variante de LK avec règle d'introduction du $\vee$ multiplicative :

$$\frac{}{\Gamma, A \vdash \Delta, A}\ Ax \text{ si } A \text{ est atomique}$$

$$\frac{\Gamma, A \to B \vdash \Delta, A \quad \Gamma, A \to B, B \vdash \Delta, C}{\Gamma, A \to B \vdash \Delta, C} \to_g \qquad \frac{\Gamma, A \vdash \Delta, A \to B, B}{\Gamma \vdash \Delta, A \to B} \to_d$$

$$\frac{\Gamma, A \wedge B, A \vdash \Delta, C}{\Gamma, A \wedge B \vdash \Delta, C} \wedge_g^1 \qquad \frac{\Gamma, A \wedge B, B \vdash \Delta, C}{\Gamma, A \wedge B \vdash \Delta, C} \wedge_g^2 \qquad \frac{\Gamma \vdash \Delta, A \wedge B, A \quad \Gamma \vdash \Delta, A \wedge B, B}{\Gamma \vdash \Delta, A \wedge B} \wedge_d$$

$$\frac{\Gamma, A \vee B, A \vdash \Delta, C \quad \Gamma, A \vee B, B \vdash \Delta, C}{\Gamma, A \vee B \vdash \Delta, C} \vee_g \qquad \frac{\Gamma \vdash \Delta, A \vee B, A}{\Gamma \vdash \Delta, A \vee B} \vee_d^1 \qquad \frac{\Gamma \vdash \Delta, A \vee B, B}{\Gamma \vdash \Delta, A \vee B} \vee_d^2$$

Les règles du calcul LKQ* sont donc celles de LK*, avec la contrainte supplémentaire que sur la gauche de la règle d'introduction gauche de la flèche, ne sont autorisées que la règle d'axiome et les règles d'introduction droite.

5.3.3 Correspondance bijective entre stratégies gagnantes pour les E-dialogues classiques et les preuves de LKQ

La correspondance entre règles et coups est la même que pour LJQ*. Le fait qu'il y ait plusieurs formules à droite signifie seulement que le proposant est autorisé à répondre à n'importe quelle formule précédemment attaquée par l'opposant, et cela, autant de fois que voulu.

Théorème 2 *Les P-stratégies gagnantes pour un E-dialogue classique sont en bijection avec les preuves de LKQ* et les arbres sous-jacents sont isomorphes.*

La preuve procède comme pour le cas intuitionniste. Si $\vec{d} : q$ avec q prouvant $\Gamma \vdash \Delta$, au lieu de définir $\sigma_{\vec{d}}$ et $a_{\vec{d}}$, on définit deux fonctions $\sigma_{\vec{d}}$ et $\rho_{\vec{d}}$, associant des entiers naturels impairs (ou bien 0) aux noms utilisés respectivement dans Γ et Δ.

En particulier, on pose :

- $\sigma_\epsilon = \emptyset$

  $\rho_\epsilon = 0$

- $\sigma_{\vec{d}([,\wedge_1)} = \sigma_{\vec{d}([,\wedge_2)} = \sigma_{\vec{d}([,\vee)} = \sigma_{\vec{d}}$

  $\rho_{\vec{d}([,\wedge_1)} = \rho_{\vec{d}([,\wedge_2)} = \rho_{\vec{d}([,A)} = \rho_{\vec{d}} \cup \{x \mapsto 2|\vec{d}|+1\}$ où x est, chaque fois, le nom de la ou des sous-formules de la formule introduite par l'inférence de tête de q

  $\rho_{\vec{d}([,\vee)} = \rho_{\vec{d}} \cup \{x_1 \mapsto 2|\vec{d}|+1\} \cup \{x_2 \mapsto 2|\vec{d}|+1\}$ où x_1 (resp x_2) est le nom de la sous-formule gauche (resp droite) de la formule disjonctive introduite par l'inférence de tête de q

- $\sigma_{\vec{d}([,A)} = \sigma_{\vec{d}(],A)} = \sigma_{\vec{d}} \cup \{x \mapsto 2|\vec{d}|+1\}$ où x est, chaque fois, le nom de A dans $\Gamma, A \vdash B$

  $\rho_{\vec{d}(],A)} = \rho_{\vec{d}}$

Aussi, si $\vec{d} \in D(\phi)$ on lui associe cette fois un séquent classique $\Gamma_{\vec{d}} \vdash \Delta_{\vec{d}}$:

- $\Gamma_\epsilon = \emptyset \qquad \Delta_\epsilon = F$
- $\Gamma_{\vec{d}(],A)} = \Gamma_{\vec{d}}, A \qquad \Delta_{\vec{d}(],A)} = \Delta_{\vec{d}}$
- $\Gamma_{\vec{d}([,\wedge_1)} = \Gamma_{\vec{d}} \qquad \phi(\vec{d})$ doit être une réponse affirmant une formule $A \wedge B$ et on pose $\Delta_{\vec{d}([,\wedge_1)} = \Delta_{\vec{d}}, A$

- $\Gamma_{\vec{d}([,\wedge_2)} = \Gamma_{\vec{d}}$ $\phi(\vec{d})$ doit être une réponse affirmant une formule $A \wedge B$ et on pose $\Delta_{\vec{d}([,\wedge_2)} = \Delta, B$
- $\Gamma_{\vec{d}([,\vee)} = \Gamma_{\vec{d}}$ $\phi(\vec{d})$ doit être une réponse affirmant une formule $A \vee B$ et on pose $\Delta_{\vec{d}([,\vee)} = \Delta, A, B$
- $\Gamma_{\vec{d}([,A)} = \Gamma_{\vec{d}}, A$ $\phi(\vec{d})$ doit être une réponse affirmant une formule $A \to B$ et on pose $\Delta_{\vec{d}([,A)} = \Delta, B$

Pour les noms associés, a chaque nouvelle formule ajoutée à $\Gamma_{\vec{d}}$ ou à $\Delta_{\vec{d}}$, on associe l'entier $|\vec{d}|+1$ (sauf pour le cas $\vec{d}([,\vee)$ où on associe à A le nom formé de la paire $(|\vec{d}|+1,1)$ et à B le nom formé de la paire $(|\vec{d}|+1,2)$.

Remarque : A chaque étape, si la règle considérée prouve un séquent $\Gamma \vdash \Delta$, alors les propriétés suivantes sont satisfaites :

- les seules P-réponses justifiées ont la forme $(m,],A)$ où A est une formule de Δ
- les seules P-attaques justifiées ont la forme $(m,[,s)$ où s attaque une formule C in Γ

5.4 Remarques

Les D-dialogues de Felscher

Les D-dialogues, définis par Felscher dans [17], diffèrent des E-dialogues par le fait que la contrainte pour l'opposant de répondre systématiquement au dernier coup du proposant est relâchée mais qu'en revanche, on lui interdit d'attaquer plusieurs fois la même affirmation du proposant.

Un D-dialogue consiste ainsi en un enchevêtrement de E-dialogues, pour lesquels l'opposant à le droit de revenir une fois en arrière, mais pas plus, sur un coup précédent.

Une P-stratégie gagnante pour un E-dialogue s'étend directement en une P-stratégie gagnante pour un D-dialogue (cf Felscher [18]).

Composition de stratégies gagnantes

Maintenant on peut donc considérer une preuve de A, voir cette preuve comme une P-stratégie gagnante pour un E-dialogue sur A et faire la même chose avec une preuve de $A \to B$ de manière à obtenir une P-stratégie gagnante pour un E-dialogue sur $A \to B$. Après le premier coup, cette dernière P-stratégie va soit attaquer sur A, soit défendre sur B. Si on met en interaction toutes ces attaques sur A avec les coups de défense de A par la preuve de A, et qu'on filtre en ne gardant que les coups de défense de B, on obtient une P-stratégie gagnante pour un jeu ouvert sur B. C'est le procédé décrit dans le cadre du "calcul de Novikoff" par Coquand dans [9] et repris au chapitre suivant.

Comme il est fait au chapitre suivant dans le cadre du calcul de Novikoff, on peut montrer que cette "interaction" suit en pas à pas le procédé d'élimination d'une coupure entre $\vdash A$ et $A \vdash B$ ou l'on réduit systématiquement la coupure la plus externe d'abord.

Chapitre 6

Débat et élimination faible de tête

Dans cette partie, nous considérons le calcul utilisé par Coquand dans [9] pour définir une alternative en terme de jeux à l'élimination des coupures. C'est un calcul des séquents classique avec formules alternativement conjonctives et disjonctives construites sur des atomes décidables. Il n'y a ni variables propositionnelles, ni implication explicite. En revanche, les conjonctions et disjonctions autorisent un nombre infini de prémisses, de telle sorte que le fragment *clos* d'une logique comme l'ω-logique y apparaît comme fragment.

Les preuves sans coupures (et sans redondances, cf définition dans le corps du chapitre) seront simplement interprétées par des stratégies gagnantes pour certains jeux à deux joueurs. L'*interaction* entre deux telles stratégies gagnantes définit un *débat*. Ce débat constitue une approche "dynamique" de l'élimination d'une coupure entre deux preuves sans coupure. Prouvé terminant par Coquand, le procédé d'interaction est comparé à l'approche habituelle d'élimination des coupures. On montre que les étapes successives d'évolution du débat correspondent exactement aux étapes successives du procédé habituel d'élimination des coupures, mais en suivant une stratégie de réduction bien précise, que, par analogie avec ce qui se passe dans le λ-calcul, on peut appeler réduction faible de tête.

6.1 Les règles infinitaires

6.1.1 L'utilisation implicite d'une règle infinitaire chez Gentzen

Sous le nom d'énonçabilité d'une règle de réduction, Gentzen [21] utilise implicitement dans sa première démonstration de cohérence de l'arithmétique, une notion de prouvabilité basée sur une règle d'introduction de la quantification universelle ayant une infinité de prémisses (ω-règle).

Considérons pour simplifier le cas de formules prénexes closes et le cas de séquents monolatères, i.e. avec des formules uniquement sur la droite du symbole $\vdash$ (ce symbole pouvant alors être omis). La notion d'"énonçabilité d'une règle de réduction" s'applique aux séquents ayant au plus une formule quantifiée universellement.

- Un séquent de la forme $\Gamma, \forall x.P(x)$ avec Γ ne contenant que des formules existentielles est réductible : une étape élémentaire de réduction consiste à choisir arbitrairement un entier n et à remplacer la formule $\forall x.P(x)$ par $P(n)$ dans le séquent.
- Un séquent Γ composé uniquement de formules existentielles est réductible : une étape élémentaire de réduction consiste à choisir une des formules $\exists x.P(x)$ du séquent ainsi qu'un n et à ajouter la formule $P(n)$ au séquent.

Une règle de réduction pour un séquent est une méthode qui, quelquesoient les n choisis arbitrairement comme instances des formules $\forall x.P(x)$, dit comment choisir un n adéquat pour les formules $\exists x.P(x)$ de telle sorte qu'au bout d'un temps fini, on obtienne un séquent dont une des formules est une proposition décidable vraie.

Autrement dit, la notion d'énonçabilité d'une règle de réduction revient à la notion suivante de prouvabilité (Γ ne contient que des formules existentielles) :

- si la proposition P est vraie alors Γ, P est prouvable
- si $\Gamma, \exists x.P(x), P(n)$ est prouvable pour un certain n alors $\Gamma, \exists x.P(x)$ est prouvable
- si, pour tout n, $\Gamma, P(n)$, est prouvable alors $\Gamma, \forall x.P(x)$ est prouvable

Toute formule prouvable de l'arithmétique est prouvable en ce dernier sens, mais, en contraste avec l'arithmétique où l'utilisation de l'axiome d'induction casse la propriété de la sous-formule, pour cette autre définition de la prouvabilité en arithmétique, la propriété de la sous-formule, de laquelle découle la cohérence, est préservée.

C'est cette notion de prouvabilité que nous considérons ci-après. Comme suggéré par la notion de "règle de réduction", cette définition de la prouvabilité se prête particulièrement bien à une approche calculatoire en termes de coups instanciant les variables quantifiées.

Remarque : Quoique non explicité par Gentzen, il est direct de montrer, à l'aide de cette définition de séquent prouvable sans coupure avec règle d'introduction du $\forall$ infinitaire, que les énoncés Σ_1^0 classiquement prouvables sont aussi intuitionnistiquement prouvables.

6.1.2 Le calcul de Novikoff

Il est remarquable de voir chez Novikoff, en 1941, la définition explicite d'un calcul propositionnel infinitaire (de type hilbertien) dont les formules A prouvables vérifient une propriété de "régularité" similaire à celle "d'énonçabilité d'une règle de réduction" pour le séquent $\vdash A$ chez Gentzen. Par un argument du même ordre que celui de Gentzen, Novikoff montre la cohérence de l'arithmétique mais aussi explicitement que les énoncés Σ^0_1 classiquement prouvables sont intuitionnistiquement prouvables.

La notion de régularité diffère de la notion d'énonçabilité d'une règle de réduction en ce léger point : une fois qu'un certain n a été choisi comme instance d'une formule $\exists x.P(x)$ il ne peut plus être choisi une nouvelle fois dans la même formule.

En plus de la notion de prouvabilité implicitement considérée par Gentzen, nous considèrerons aussi la notion de prouvabilité implicite à la notion de "régularité" de Novikoff. Nous montrerons en plus en quoi la restriction supplémentaire imposée par Novikoff ne fait rien perdre à la prouvabilité.

6.1.3 Règles infinitaires chez Lorenzen

Du côté occidental, selon Schütte, c'est chez Lorenzen [36] en 1951, qu'on rencontre la première utilisation explicite d'une règle avec une infinité de prémisses. L'idée est la même et le but est le même que chez Gentzen : prouver un théorème d'existence de preuves normales duquel s'ensuit la cohérence du système considéré. Dans le cas de Lorenzen, c'est d'un sous-système de la théorie des types de Russell qu'il s'agit.

6.1.4 Formules et boréliens

A noter aussi une analogie faite par Martin-Löf [41] entre formules construites à partir de conjonctions et disjonctions infinies et boréliens, analogie qui permet de prouver constructivement la transitivité de l'inclusion entre boréliens à l'aide de l'élimination des coupures.

6.2 Une variante du calcul infinitaire de Gentzen

Nous reprenons le calcul implicite à la notion d'énonçabilité d'une règle de réduction, le généralisant au cas d'une logique propositionnelle infinitaire sans variables propositionnelles, les atomes se limitant aux propositions "vrai" et "faux". On y ajoute aussi une règle de coupure explicite.

Nous introduisons une notion de "redondance" signifiant qu'un même n est joué plusieurs fois dans une formule disjonctive. Le sous-calcul sans coupure et sans "redondances" du premier calcul que nous décrivons correspondra au calcul implicite à la notion de régularité de Novikoff.

6.2.1 Formules et séquents

Les formules sont des arbres bien-fondés à branchement alternativement disjonctif et conjonctif. Aux feuilles, on trouve soit la formule vraie **1** soit la formule fausse **0**.

- si $(A_i)_{i\in I}$ est une famille de formules conjonctives alors $\vee_{i\in I}A_i$ est une **formule disjonctive**
- si $(A_i)_{i\in I}$ est une famille de formules disjonctives, alors $\wedge_{i\in I}A_i$ est une **formule conjonctive**

On définit la formule vraie **1** comme étant la famille vide de formule disjonctive et la formule fausse **0** comme étant la famille vide de formule conjonctive. Ainsi **1** est conjonctive et **0** est disjonctive.

Remarque : On peut, a priori, choisir des ensembles d'indexation quelconques. Dans la pratique, par exemple lorsqu'on interprète les formules du calcul des prédicats ou celles de l'arithmétique, les ensembles d'indexation sont des ensembles d'entiers.

La **formule duale** de A, notée $A^\perp$ est inductivement définie par :

- $(\vee_{i\in I}A_i)^\perp = \wedge_{i\in I}A_i^\perp$

- $(\wedge_{i\in I} A_i)^\perp = \vee_{i\in I} A_i^\perp$

En particulier, on a $\mathbf{1}^\perp = \mathbf{0}$ et $\mathbf{0}^\perp = \mathbf{1}$

La notion d'occurrence d'un sous-arbres et d'adresse de cette occurrence est une notion standard pour les arbres. Ici, une adresse dans une formule définit une sous-formule de cette formule. Nous utilisons les lettres u, v, w, ... pour noter les adresses de formules.

- $<>$ est l'**adresse** de A en tant que sous-formule d'elle-même
- si u est l'adresse d'une occurrence de la sous-formule $\vee_{i\in I} A_i$ de A alors ui est une **adresse** de la sous-formule A_i de A
- si u est l'adresse d'une occurrence de la sous-formule $\wedge_{i\in I} A_i$ de A alors ui est une **adresse** de la sous-formule A_i de A

Si v a la forme $ui_1...i_n$ alors v est une **extension** de u. Si v a la forme ui alors v est appelée **extension directe** de u.

Nous considérons un ensemble infini $\mathcal{N}$ appelé **ensemble des noms**. Nous considérons aussi une fonction $\mathcal{F}$ qui associe à chaque sous-ensemble X de $\mathcal{N}$ un nom x qui n'est pas dans X. Une **fonction de renommage** σ est une bijection d'un ensemble $X \subset \mathcal{N}$ vers un autre ensemble $Y \subset \mathcal{N}$. La fonction de renommage de domaine vide est notée $\{\}$. L'adjonction d'un élément et son image à une fonction de renommage est notée sous la forme $\sigma \cup \{t \to t'\}$

Un **séquent** est relativisé à une formule A donnée. Un séquent est un ensemble fini d'adresses d'occurrences de sous-formules disjonctives, chacune indexée par un nom de $\mathcal{N}$, avec, en plus, éventuellement l'adresse — non nommée — d'une occurrence d'une sous-formule, cette fois, conjonctive. Une occurrence de sous-formule disjonctive peut se rencontrer plusieurs fois, mais son adresse doit figurer chaque fois avec un nom différent.

Il est commode, ici encore, de considérer des séquents avec bénitier. Le bénitier sera vide ou rempli selon que le séquent contient l'adresse d'une formule conjonctive ou pas (ladite adresse prenant place dans le bénitier).

Les majuscules grecques Γ, Δ sont réservées pour noter des ensembles d'adresses nommées d'occurrences de sous-formules disjonctives de A. Un séquent s'écrira alors sous la forme Γ (cas où le bénitier est vide) ou $\Gamma; B$ (cas où le bénitier contient l'adresse d'une occurrence d'une certaine sous-formule conjonctive B).

Une écriture comme Γ, A, Δ suppose implicitement qu'à la place de A, c'est l'adresse d'une de ses occurrences que l'on trouve et que cette adresse est nommée avec un nom qui n'interfère pas avec les autres noms apparaissant dans Γ ou Δ, eux-mêmes ne partageant aucun de leurs noms.

Une notation explicite pour indiquer les noms est celle-ci : Γ^X, A^z, Δ^Y signifiant que X rassemble les noms des adresses des occurrences des formules de Γ, que Y rassemble ceux de Δ et que A a le nom z. Quant à la notation $[\Gamma_i]_i$, elle représente une famille de séquents.

6.2.2 Preuves

Une **preuve** est un arbre bien-fondé défini inductivement au moyen des règles d'inférence suivantes :

$$\frac{\Gamma^X, (\vee_{i\in I} A_i)^x; A_{i_0}}{\Gamma^X, (\vee_{i\in I} A_i)^x;}\ I_\vee \qquad (i_0 \in I)$$

$$\frac{\left[\Gamma^X, A_j^z;\right]_{j\in J}}{\Gamma^X; \wedge_{j\in J} A_j}\ I_\wedge$$

$$\frac{\Gamma_1^{X_1}, \Delta^{X_0}; A \quad \Gamma_2^{X_2'}, \Delta^{X_0}, A^x;}{\Gamma_1^{X_1}, \Gamma_2^{X_2}, \Delta^{X_0};}\ Coupe \qquad (X_2 = \sigma(X_2'))$$

Notons qu'un cas particulier de la règle $I_\wedge$ est la règle suivante :

$$\overline{\Gamma; \mathbf{1}}$$

ce qui assure l'existence effective de preuves (i.e. d'arbres bien-fondés).

Les règles $I_\vee$ et $I_\wedge$ sont des règles d'introduction et la règle *Coupe* une règle de coupure. Si p est une preuve, on note $|p|$ son séquent conclusion.

Remarques : 1) Comme signalé au chapitre 1, la présence d'un contexte Δ partagé par les deux prémisses et de contextes Γ_1 et Γ_2 non partagés permet d'éviter le recours à des règles de contraction ou d'affaiblissement au cours du processus d'élimination des coupures.

2) Pour la règle $I_\wedge$, bien que les A_j soient des formules a priori distinctes, on leur donne, par commodité, dans chaque séquent conclusion des prémisses, le même nom z.

3) Pour la règle *Coupe*, l'ensemble X_2' des noms utilisés pour Γ_2 dans le séquent conclusion de la prémisse de droite est intentionnellement supposé différent de l'ensemble X_2 des noms utilisés pour Γ_2 dans le séquent conclusion de la coupure (ceci pour des raisons qui apparaissent au moment d'éliminer les coupures). On associe alors à la coupure une fonction de renommage, disons σ, de X_2' vers X_2.

Remarquons aussi que la contrainte du bénitier impose que l'inférence de tête de la prémisse de la règle $I_\vee$ soit une règle $I_\wedge$ introduisant la formule A_{i_0}. On rencontre donc toujours la règle $I_\vee$ dans cette configuration :

$$\cfrac{\cfrac{[\Gamma, (\vee_{i\in I} \wedge_{j\in J} A_{ij})^x, (A_{i_0 j})^z;]_{j\in J}}{\Gamma, (\vee_{i\in I} \wedge_{j\in J} A_{ij})^x; \wedge_{j\in J} A_{ij}}\, I_\wedge}{\Gamma, (\vee_{i\in I} \wedge_{j\in J} A_{ij})^x;}\, I_\vee \qquad (i_0 \in I)$$

De même, on rencontrera toujours la règle de coupure dans cette configuration :

$$\cfrac{\cfrac{\left[\Gamma_1^{X_1}, \Delta^{X_0}, (A_j)^z;\right]_{j\in J}}{\Gamma_1^{X_1}, \Delta^{X_0}; \wedge_{j\in J} A_j}\, I_\wedge \qquad \Gamma_2^{X_2'}, \Delta^{X_0}, A^x;}{\Gamma_1^{X_1}, \Gamma_2^{X_2}, \Delta^{X_0};}\, Coupe$$

Dans la suite, nous simplifierons les notations en laissant tomber l'information des ensembles parcourus par les indices et des noms utilisés dans les séquents. En revanche, l'information des noms intervient dans la notation des preuves par des termes.

6.2.3 Notation des preuves par des termes

Nous proposons une notation alternative des preuves sous la forme de termes.

Dans le cas de la règle $I_\vee$, si la preuve du séquent $\Gamma, (\vee_i A_i)^x; A_{i_0}$ est p_0 alors la preuve de $\Gamma, (\vee_i A_i)^x$ est notée :

$$!(x, i_0)p_0$$

Dans le cas de la règle $I_\wedge$, si les preuves des $\Gamma, (A_j)^z$ sont les p_j, alors la preuve de $\Gamma; \wedge_j A_j$ est notée :

$$?j \overset{z}{\mapsto} p_j$$

Dans le cas de la règle *Coupe*, si on pose $X = X_1 \cup X_2 \cup X_0$ tandis que les preuves de $\Gamma_1^{X_1}, \Delta^{X_0}; \wedge_j A_j$ et de $\Gamma_2,^{X_2'} \Delta^{X_0}, (\vee_j A_j^\perp)^x$ sont respectivement p et q alors, la preuve de $\Gamma_1^{X_1}, \Gamma_2^{X_2}, \Delta^{X_0}$ est notée :

$$\mathrm{Coupe}_{(x,X,\sigma)}(p; q)$$

La donnée du séquent conclusion d'une preuve et de sa notation comme un terme permet de retrouver les séquents prouvés à chaque étape. Dans la suite, on considèrera que le séquent prouvé par une preuve notée comme un terme est à chaque instant connu, sans se soucier de savoir comment il est connu.

La donnée d'un nom x et d'un indice i_0, écrite sous la forme x, i_0 est appelée $\vee$-**coup**. Si p est une preuve de la forme $!(x, i_0)p_0$ alors x, i_0 est appellé $\vee$-**coup** de p et noté $C(p)$. Si x est le nom d'une adresse u, alors on pose $\mathcal{O}cc(x, i_0) = ui_0$.

6.2.4 Preuves de Novikoff

Un séquent est **avec redondance** s'il contient deux fois la même occurrence d'une formule, les adresses étant bien sûr indexées chaque fois d'un nom différent. Dans le cas contraire, le séquent est dit **sans redondance**. Une preuve est dite **sans redondance** s'il n'y intervient que des séquents sans redondance. Une **preuve de Novikoff** est une preuve sans coupure et sans redondance. Autrement dit, une preuve de Novikoff est construite à partir des règles suivantes :

$$\frac{\Gamma, \vee_{i \neq i_0} A_i ; A_{i_0}}{\Gamma, \vee_i A_i ;} \; I'_\vee$$

et

$$\frac{[\Gamma, A_j;]_j}{\Gamma; \wedge_j A_j} \; I_\wedge$$

et on retrouve la notion de prouvabilité implicite à la notion de régularité de Novikoff.
Remarque : 1) L'interdiction de redondance n'enlève rien à la prouvabilité. De fait, la règle suivante est admissible :

$$\frac{\Gamma, (\vee_i A_i)^x, (\vee_i A_i)^y ;}{\Gamma, (\vee_i A_i)^x ;} \; Cont$$

La démonstration d'admissibilité est par induction sur la preuve de $\Gamma, (\vee_i A_i)^x, (\vee_i A_i)^y$ supposée elle-même sans redondance. Le cas clé est quand $\Gamma, (\vee_i A_i)^x, (\vee_i A_i)^y;$ provient de $\Gamma, (\vee_i A_i)^x, (\vee_i A_i)^y; A_{i_0}$ par une règle $I_\vee$ appliquée à la formule de nom y. Par induction, on a l'existence d'une preuve de $\Gamma, (\vee_i A_i)^x; A_{i_0}$ et il suffit de réappliquer la règle $I_\vee$, cette fois avec le nom x.

2) Notre définition est plus faible que celle de Novikoff. De fait, chez Novikoff, on peut faire intervenir des variables propositionnelles dans les formules et on a, en plus, des axiomes du style $\Gamma, A^\perp, A$.

3) Les preuves de Novikoff ne sont pas stables par élimination des coupures, des redondances pouvant y survenir.

6.2.5 Preuves partielles

Dans le but de pouvoir étudier une version épurée de la coupure, à savoir :

$$\frac{;A \quad A^\perp;}{} \; Coupe$$

nous considérons aussi des preuves partiellement bien définies. A cette fin, nous ajoutons la règle d'inférence suivante à la définition de preuve :

$$\frac{}{\Gamma} \; \Omega$$

quelque soit Γ, la notation sous forme de terme d'une telle preuve étant Ω.

6.3 Interprétation en terme de jeux

6.3.1 Parties

On définit ci-dessous un jeu entre deux joueurs, le premier étant appelé proposant et le second opposant. Les joueurs se disputent la validité d'une certaine formule, disons A, et leurs coups s'alternent. Un coup du proposant (resp opposant) consiste en l'affirmation d'une sous-formule conjonctive de A (resp $A^\perp$) qui est en même temps une attaque d'une sous-formule de $A^\perp$ (resp A) précédemment affirmée par l'opposant (resp proposant). La formule attaquante doit être une sous-formule de la duale de la formule attaquée. Une partie est une suite finie ou infinie de coups (sous-entendu de coups permis) dont le coup initial dépend de la nature de A. Si A est disjonctive, le coup initial est une affirmation de $A^\perp$ par l'opposant. Sinon, c'est l'affirmation, toujours par l'opposant, d'une sous-formule conjonctive directe de $A^\perp$. On représente un coup par la donnée (f, u) d'un entier naturel f et d'une adresse u dans A (ce sont les mêmes adresses que dans $A^\perp$). L'adresse u désigne l'occurrence d'une sous-formule de A et l'entier f est censé désigner l'indice du coup duquel u est une attaque. Ainsi, si le coup d'indice f était (f', u'), alors u désigne une sous-formule de la formule d'adresse u', i.e. u est une extension directe de u'. Puisque le coup initial affirme $A^\perp$, il a la forme $(f, <>)$ et puisqu'il n'attaque aucun coup précédent, on peut se permettre de poser arbitrairement $f = 0$.

En résumé :

Un **J-coup** pour A est la donnée (f, u) d'un entier naturel f et d'une occurrence u dans A.

Une **partie** π pour A est une suite non vide finie ou infinie de J-coups $(f_0, u_0), ..., (f_n, u_n), ...$ telle que $f_0 = 0$, $u_0 =<>$ et, pour tout $n > 0$, $f_n < n$ et u_n est une extension directe de u_{f_n}. Si π est la partie finie $(f_0, u_0), ..., (f_n, u_n)$, alors, pour tout $m < n$, $\pi|_m$ désigne la partie $(f_0, u_0), ..., (f_m, u_m)$.

Un type particulier de partie sont les E-parties. Pour les E-parties sur A, on a une bijection entre les stratégies gagnantes du proposant et les preuves de Novikoff de A.

Une E-partie est une partie $(f_0, u_0), ..., (f_n, u_n), ...$ telle que $f_n = n - 1$ pour n pair. Autrement dit, une E-partie est une partie où l'opposant répond systématiquement au coup précédent du proposant. De plus, il est défendu au proposant de rejouer deux fois la même formule ré-attaquant le même coup précédent de l'opposant.

Une E-partie est gagnante pour le proposant si l'opposant ne peut plus jouer, c'est-à-dire si le dernier coup du proposant affirme la formule **1**.

6.3.2 Stratégies

On définit dans cette section, les stratégies du proposant pour les E-parties. Une telle stratégie peut être définie inductivement par la donnée de l'ensemble des suites de coups joués par l'opposant au fur et à mesure que le proposant joue les coups de sa stratégie. On appelle position les suites de coups de l'opposant. Notons que pour une E-partie, l'opposant répond toujours au coup précédent. On peut donc se dispenser de l'indice f et ne garder que la composante u des coups (f, u). Précisons aussi que puisque les formules affirmées par l'opposant sont conjonctives dans $A^\perp$, alors, elles sont disjonctives dans A.

Soit A une formule.

Une **position** sur A est une suite non vide $u_1...u_n$ d'occurrences de sous-formules conjonctives de $A^\perp$ (i.e. disjonctives de A) telle que :

- si A est disjonctive, $<>$ est une **position** (de longueur 1)
- si $A = \wedge_{i\in I} A_i$, pour chaque $i \in I$, i est une **position** (de longueur 1)
- $u_1...u_{n+1}$ est une **position** si $u_1...u_n$ est une position, et s'il existe un entier $m \leq n$ tel que u_{n+1} s'écrive $u_m ij$, mais qu'aucun n'autre $u_{m'}$ ne s'écrive $u_m ij'$ avec le même i

Remarque : La condition qu'aucun autre $u_{m'}$ ne s'écrive $u_m ij'$ avec le même i correspond à la condition que le proposant n'a pas le droit de rejouer deux fois le même coup. Une conséquence est que les $u_{m'}$ sont distincts les uns des autres, et en particulier que u_m est unique vérifiant $u_{n+1} = u_m ij$.

Une **stratégie** (sous-entendu du proposant) pour A est une fonction ϕ définie sur un ensemble $\mathrm{Dom}(\phi)$ de positions et à valeurs dans l'ensemble des occurrences de sous-formules conjonctives de A. Une stratégie doit satisfaire en outre ces trois propriétés :

1. $\mathrm{Dom}(\phi)$ contient les positions de longueur 1
2. si $\phi(u_1...u_n) = v$ fait référence à la sous-formule $\wedge_i A_i$, alors, pour tout i, posant $u_{n+1} = vi$, on a $u_1...u_{n+1} \in \mathrm{Dom}(\phi)$
3. si $u_1...u_{n+1} \in \mathrm{Dom}(\phi)$ alors $u_1...u_n \in \mathrm{Dom}(\phi)$, u_{n+1} est une extension directe de $v = \phi(u_1...u_n)$ et v est une extension directe de l'un des u_m

Une stratégie ϕ est **gagnante** en $u_1...u_n \in \mathrm{Dom}(\phi)$ si, $v = \phi(u_1...u_n)$ faisant référence à la sous-formule $\wedge_i A_i$, pour tout $i \in I$, posant $u_{n+1} = vi$, ϕ est gagnante en $u_1...u_{n+1}$.

En particulier, si $\phi(u_1...u_n)$ fait référence à $\mathbf{1}$ alors ϕ est gagnante en $u_1...u_n$, conformément à la définition de partie gagnante pour le proposant.

Une stratégie ϕ est **gagnante** pour A si elle est gagnante en toute position de longueur 1.

Remarque : Une stratégie pour une formule conjonctive $\wedge_i A_i$ est gagnante si elle est gagnante pour chaque A_i.

6.3.3 Equivalence entre stratégies gagnantes et preuves de Novikoff

Il y a une bijection directe entre preuves de Novikoff de A et stratégies gagnantes du proposant pour A. Nous le prouvons par un argument similaire à celui développé au chapitre 5 à propos des E-dialogues.

Soit p une preuve de Novikoff de A. L'ensemble des occurrences de p vu comme un arbre a la structure d'un ensemble de positions. Par induction sur la taille de ces occurrences, on associe à chacune de ces occurrences une position. Par construction, les positions sont toutes distinctes.

- on associe la position $<>$ à p en tant que sous-preuve d'elle-même ($<>$ est une position de longueur 1)
- si $A = \wedge_i A_i$, et $p = ?i \mapsto p_i$, alors on associe la position i à chacun des p_i
- si la position associée à la sous-preuve q est $u_1...u_n$, si $q = !(x, i_0)?j \mapsto q_j$ et si $\mathcal{O}cc(x, i_0) = v$ alors, posant $u = vj$, on associe à q_j la position $u_1...u_n u$

L'absence de redondance implique que les suites $u_1...u_n$ sont effectivement des positions et on définit le domaine $\mathrm{Dom}(\phi)$ de la stratégie ϕ associée à p comme l'ensemble des positions associées aux sous-preuves de p. Maintenant, si la sous-preuve de p correspondant à la position $u_1...u_n \in \mathrm{Dom}(\phi)$ a la forme $!(x, i_0)?j \mapsto q_j$ alors on pose $\phi(u_1...u_n) = \mathcal{O}cc(x, i_0)$. Il est immédiat de vérifier que ϕ satisfait les trois propriétés qu'une stratégie doit satisfaire.

Pour la réciproque, une étape préliminaire est de nommer chaque occurrence de sous-formule de A. Ensuite, à toute position $u_1...u_n$ on associe un séquent $\Gamma(u_1...u_n)$ constitué des occurrences u_1, ..., u_n, chacune avec le nom qui lui est propre. La propriété que tous les u_m soient distincts entre eux implique que ce séquent est sans redondance.

Maintenant, à toute position $u_1...u_n$, on peut associer une preuve de Novikoff de $\Gamma(u_1...u_n)$.

Soit donc $u_1...u_n$ une position telle que $\phi(u_1...u_n) = u_m i_0$. Supposons que u_m ait le nom x dans $\Gamma(u_1...u_n)$, que $u_m i_0$ désigne la sous-formule $\wedge_j B_j$ et que, pour tout $j \in J$, posant $u_{n+1} = u_m i_0 j$, on ait associé à $u_1...u_{n+1}$ une preuve q_j de $\Gamma(u_1...u_{n+1})$. Alors, par induction bien-fondée sur la structure de $\mathrm{Dom}(\phi)$, on peut associer à $u_1...u_n$ la preuve $!(x, i_0)?j \mapsto q_j$ qui est une preuve de Novikoff de $\Gamma(u_1...u_n)$.

Comme dans le cas des E-dialogues du chapitre 5, ces deux traductions sont réciproques l'une de l'autre et constituent dont une correspondance bijective.

6.3.4 Stratégies partielles

Les preuves partielles sont associées à des stratégies partielles.

Si on retire la condition 1 de la définition de stratégie, on obtient la définition de **stratégie partielle**.

Une stratégie partielle ϕ est **non-perdante** en $u_1...u_n$ soit parce que $u_1...u_n \notin \mathrm{Dom}(\phi)$ soit, au cas où $u_1...u_n \in \mathrm{Dom}(\phi)$, parce que pour tout $i \in I$, elle est non-perdante en $u_1...u_{n+1}$, où l'on a posé $u_{n+1} = vi$, v étant l'occurrence de la sous-formule $\wedge_{i \in I} A_i$.

La correspondance entre preuves de Novikoff et stratégies gagnantes s'étend au cas des preuves partielles de Novikoff et des stratégies partielles et non-perdantes.

6.3.5 Débat

Soit ϕ une stratégie partielle non-perdante du proposant pour un jeu sur $A = \vee_i A_i^\perp$ et τ une stratégie partielle non-perdante du proposant pour un jeu sur $A^\perp = \wedge_i A_i$.

On va définir une partie obtenue en faisant interagir entre eux les proposants des jeux pour A et pour $A^\perp$. Cette partie n'est plus une E-partie.

Les définitions suivantes proviennent, à peu de choses près, de Coquand [9].

A toute partie finie, on associe une position courante. La **position courante** l_n d'une partie finie $\pi = (f_0, u_0), ..., (f_n, u_n)$ est définie par induction bien-fondée à partir de (f_n, u_n) et des positions courantes l_i des sous-parties $\pi|_i$ pour $i < n$.

- $l_0 =<>$ (position restreinte à la seule occurrence $<>$)
- si $f_n = 0$ alors $l_n = u_n$
- si $l_{f_n - 1} = u_{n_1}...u_{n_k}$ alors $l_n = u_{n_1}...u_{n_k}u_n$

Remarques : 1) Les entiers impairs correspondent aux coups de ϕ et les entiers pairs aux coups de τ. L'entier f_n est l'index du coup auquel répond le $n^{\text{ième}}$ coup. Ainsi, f_n et n sont de parités distinctes.

2) Si l_n est la position courante de la partie $(f_0, u_0), ..., (f_n, u_n)$ alors, on peut retrouver par induction d'où "provient" chaque occurrence de la position. Ainsi, une position courante peut toujours s'écrire sous la forme $l_n = u_{m_1}...u_{m_n}$.

Le **débat** entre ϕ et τ est une partie $(f_0, u_0), ..., (u_n, f_n), ...$ définie récursivement à partir du calcul des positions courantes successives. Pour $n \geq 0$, on considère $l_n = u_{m_0}...u_{m_n}$ la position courante associée à la partie finie $(f_0, u_0), ..., (u_n, f_n)$. Si n est pair, on pose $u_{n+1} = \phi(u_{m_1}...u_{m_n})$, si n est impair, on pose $u_{n+1} = \tau(u_{m_1}...u_{m_n})$. Dans tous les cas, u_{n+1} est une extension directe d'un unique u_{m_i} et on pose $f_{n+1} = m_i$.

Proposition 6 *Le débat entre ϕ et τ est fini, c'est-à-dire que, pour un certain n, $\phi(l_n)$ ou $\tau(l_n)$ n'est plus défini.*

On trouvera la preuve dans Coquand [9].

6.4 Elimination faible de tête des coupures

L'élimination faible de tête des coupures est une procédure de réduction itérée de la coupure de tête entre deux preuves, lorsque celle-ci existe, jusqu'à ce que la coupure laisse la place en tête à une règle d'introduction.

Avant de décrire cette procédure, on donne une liste de règle de réduction permettant de transformer les configurations ayant une coupure comme règle principale en d'autres configurations plus petite relativement à une certaine mesure bien choisie.

6.4.1 Réduction de tête élémentaire

La réduction de tête élémentaire consiste à réduire une coupure de tête en une ou plusieurs coupures plus petites relativement à une certaine mesure bien choisie.

On envisage différents cas selon l'inférence de tête de la prémisse se trouvant sur le côté droit de la coupure. On ne considère pas le cas où l'inférence de tête en question est une coupure.

T1 La règle d'introduction en tête de la prémisse de droite "joue" dans la formule de coupure (on pose $A_{i_0}^\perp = \wedge_j B_j^\perp$ et $\sigma(X_2') = X_2$).

$$\dfrac{\dfrac{\left[\begin{array}{c}\vdots\; p_i\\ \Delta^X,\Gamma_1^{X_1},A_i^y;\end{array}\right]_i}{\Delta^X,\Gamma_1^{X_1};\wedge_i A_i}\,I_\wedge \qquad \dfrac{\dfrac{\left[\begin{array}{c}\vdots\; q_j\\ \Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x,(B_j^\perp)^z;\end{array}\right]_j}{\Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x;\wedge_j B_j^\perp}\,I_\wedge}{\Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x;}\,I_\vee}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2};}\,Coupe$$

se réduit en

$$\dfrac{\dfrac{\left[\dfrac{\dfrac{\left[\begin{array}{c}\vdots\; p_i\\ \Delta^X,\Gamma_1^{X_1},A_i^y;\end{array}\right]_i}{\Delta^X,\Gamma_1^{X_1};\wedge_i A_i}\,I_\wedge \qquad \begin{array}{c}\vdots\; q_j\\ \Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x,(B_j^\perp)^z;\end{array}}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2},(B_j^\perp)^z;}\,Coupe\right]_j}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2};\wedge_j B_j^\perp}\,I_\wedge \qquad \begin{array}{c}\vdots\; p_{i_0}\\ \Delta^X,\Gamma_1^{X_1},A_{i_0}^y;\end{array}}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2};}\,Coupe$$

T2 La règle d'introduction en tête de la prémisse de droite "joue" dans une formule autre que la formule de coupure (on pose $B_{j_0}=\wedge_k C_k$ et $\sigma(X'_2)=X_2$).

$$\dfrac{\dfrac{\left[\begin{array}{c}\vdots\; p_i\\ \Delta^X,\Gamma_1^{X_1},A_i^y;\end{array}\right]_i}{\Delta^X,\Gamma_1^{X_1};\wedge_i A_i}\,I_\wedge \qquad \dfrac{\dfrac{\left[\begin{array}{c}\vdots\; q_j\\ \Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x,(\vee_j B_j)^{x'},C_k^z;\end{array}\right]_k}{\Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x,(\vee_j B_j)^{x'};\wedge_k C_k}\,I_\wedge}{\Delta^X,\Gamma_2^{X_2},(\vee_i A_i^\perp)^x,(\vee_j B_j)^{x'};}\,I_\vee}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2},\vee_j B_j;}\,Coupe$$

se réduit en

$$\dfrac{\dfrac{\left[\dfrac{\dfrac{\left[\begin{array}{c}\vdots\; p_i\\ \Delta^X,\Gamma_1^{X_1},A_i^y;\end{array}\right]_i}{\Delta^X,\Gamma_1^{X_1};\wedge_i A_i}\,I_\wedge \qquad \begin{array}{c}\vdots\; q_j\\ \Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x,(\vee_j B_j)^{x'},C_j;\end{array}}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2},(\vee_j B_j)^{x'},C_k^z;}\,Coupe\right]_k}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2},(\vee_j B_j)^{x'};\wedge_k C_k}\,I_\wedge}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2},(\vee_j B_j)^{x'};}\,I_\vee$$

T3 la preuve de droite est Ω (on pose $\sigma(X'_2)=X_2$).

$$\dfrac{\dfrac{\left[\begin{array}{c}\vdots\; p_i\\ \Delta^X,\Gamma_1^{X_1},A_i^y;\end{array}\right]_i}{\Delta^X,\Gamma_1^{X_1};\wedge_i A_i} \qquad \begin{array}{c}\Omega\\ \vdots\\ \Delta^X,\Gamma_2^{X'_2},(\vee_i A_i^\perp)^x;\end{array}}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2};}$$

se réduit en

$$\dfrac{\Omega}{\Delta^X,\Gamma_1^{X_1},\Gamma_2^{X_2};}$$

Remarque : Pour la première règle de réduction, il y a une duplication de p_i. Supposons que dans p_i, il y ait un “coup” dans une formule de Δ, Γ_1. En appliquant maintenant la deuxième règle de réduction pour chacune des copies de p_i, ce coup peut être déplacé au delà de la coupure la plus basse. Ce coup, joué une première fois par une des copies de p_i puis une deuxième fois par l'autre copie de p_i, correspondra donc à une redondance. Et il y aura cette introduction de redondance même si les preuves avaient été sans redondance au départ. C'est pourquoi nous autorisons plusieurs occurrences de la même sous-formule dans un séquent et pourquoi nous avons recours aux noms pour distinguer entre ces diverses occurrences de la même sous-formule.

En adoptant l'écriture des preuves sous la forme de termes, et en appelant $\stackrel{t}{\Longrightarrow}$ la réduction ainsi définie, les règles précédentes se réécrivent de la sorte (on pose $\overline{X} = X \cup X_1 \cup X_2$) :

T1

$$\mathrm{Coupe}_{(x,\overline{X},\sigma)}(?i \stackrel{y}{\mapsto} p_i ; !(x, i_0)?j \stackrel{z}{\mapsto} q_j) \stackrel{t}{\Longrightarrow} \mathrm{Coupe}_{(y,\overline{X},\{\})}(?j \stackrel{\mathcal{F}(\overline{X})}{\mapsto} \mathrm{Coupe}_{(x,\overline{X}\cup\mathcal{F}(\overline{X}),\sigma\cup\{z\to\mathcal{F}(\overline{X})\})}(?i \stackrel{y}{\mapsto} p_i ; q_j); p_{i_0})$$

T2

$$\mathrm{Coupe}_{(x,\overline{X},\sigma)}(?i \stackrel{y}{\mapsto} p_i ; !(x', i'_0)?j \stackrel{z}{\mapsto} q_j) \stackrel{t}{\Longrightarrow} !(\sigma(x'), i'_0)?j \stackrel{\mathcal{F}(\overline{X})}{\mapsto} \mathrm{Coupe}_{(x,\overline{X}\cup\{\mathcal{F}(\overline{X})\},\sigma\cup\{z\to\mathcal{F}(\overline{X})\})}(?i \stackrel{y}{\mapsto} p_i ; q_j)$$

T3

$$\mathrm{Coupe}_{(x,\overline{X},\sigma)}(?i \stackrel{Z_i}{\mapsto} p_i ; \Omega) \stackrel{t}{\Longrightarrow} \Omega$$

L'information, pour chaque coupure, des “$\overline{X}$” et “σ” sert à gérer les éventuels problèmes de confusion de nom, suite à des redondances. Il n'y a pas d'autres difficultés que celles prises en compte par ces règles, et, à l'avenir, par souci de lisibilité, nous omettrons ces informations.

6.4.2 Réduction de tête étendue

La réduction de tête étendue d'une preuve p consiste à aller chercher, à travers les coupures, la règle $I_\vee$ la plus externe et la plus à gauche de p, d'appliquer, à la bonne place, les règles 2 ou 3 de la réduction de tête élémentaire, jusqu'à ce que cette règle $I_\vee$, et la règle $I_\wedge$ qui suit, viennent en tête de p, auquel cas on s'arrête, ou bien que la règle 1 soit applicable, auquel cas on l'applique et on s'arrête.

On utilise comme intermédiaire la réduction $\stackrel{d}{\Longrightarrow}$ qui exprime qu'une règle d'introduction se situant sur la branche d'extrême-gauche traverse effectivement jusqu'à la tête de la preuve.

D1

$$!(x, i_0)p_0 \stackrel{d}{\Longrightarrow} !(x, i_0)p_0$$

D2

$$?j \stackrel{z}{\mapsto} p_j \stackrel{d}{\Longrightarrow} ?j \stackrel{z}{\mapsto} p_j$$

D3

$$\Omega \stackrel{d}{\Longrightarrow} \Omega$$

D4

$$\frac{q \stackrel{d}{\Longrightarrow} !(x', j_0)?k \stackrel{t}{\mapsto} q_k}{\mathrm{Coupe}_x(p; q) \stackrel{d}{\Longrightarrow} !(x', j'_0)?k \stackrel{t}{\mapsto} \mathrm{Coupe}_x(p; q_k)}$$

D5

$$\frac{q \stackrel{d}{\Longrightarrow} \Omega}{\mathrm{Coupe}_x(p; q) \stackrel{d}{\Longrightarrow} \Omega}$$

FE1

$$\frac{q \overset{f}{\Longrightarrow}_1 \text{Coupe}_y(q'; r)}{\text{Coupe}_x(p; q) \overset{f}{\Longrightarrow}_1 \text{Coupe}_x(p; \text{Coupe}_y(q'; r))}$$

FE2

$$\frac{q \overset{d}{\Longrightarrow} !(x, j_0)?k \overset{t}{\mapsto} q_k}{\text{Coupe}_x(?j \overset{z}{\mapsto} p_j; q) \overset{f}{\Longrightarrow}_1 \text{Coupe}_z(?k \overset{t}{\mapsto} \text{Coupe}_x(?j \overset{z}{\mapsto} p_j; q_k); p_{j_0})}$$

6.4.3 La réduction faible de tête

L'élimination (ou réduction) faible de tête des coupures itère la réduction de tête étendue jusqu'à ce qu'une règle d'introduction vienne en tête.

F1

$$\frac{p \overset{d}{\Longrightarrow} r}{p \overset{f}{\Longrightarrow} r}$$

F2

$$\frac{\text{Coupe}_x(p; q) \overset{f}{\Longrightarrow}_1 r \quad r \overset{f}{\Longrightarrow} s}{\text{Coupe}_x(p; q) \overset{f}{\Longrightarrow} s}$$

Si pour certains p et r, on a $p \overset{f}{\Longrightarrow} r$, alors, r est sans coupure en tête, et, par induction, on a l'existence d'une suite finie $p = p_0, p_1, ..., p_n$ telle que $p_i \overset{f}{\Longrightarrow}_1 p_{i+1}$ pour $i < n$ et $p_n \overset{d}{\Longrightarrow} r$.

La suite $p_0, p_1, ..., p_n$ est appelée **trace** de la réduction faible de tête de p.

6.4.4 Terminaison de l'élimination faible de tête des coupures

Pour prouver la terminaison de l'élimination faible de tête des coupures, nous avons besoin de prouver quelque chose de plus fort : la terminaison de l'algorithme — a priori transfini — d'élimination des coupures selon la stratégie de réduction de la coupure plus externe d'abord.

La **stratégie de réduction de la coupure la plus externe d'abord** est une stratégie de réduction qui étend la réduction faible de tête sous les règles d'introduction. A partir du moment où le langage est infinitaire, ce procédé de réduction peut être transfini.

E1

$$\frac{p \overset{f}{\Longrightarrow} !(x, i_0)p_0 \qquad p' \overset{e}{\Longrightarrow} r'}{p \overset{e}{\Longrightarrow} !(x, i_0)r'}$$

E2

$$\frac{p \overset{f}{\Longrightarrow} ?j \overset{z}{\mapsto} p_j \qquad \forall j, p_j \overset{e}{\Longrightarrow} r_j}{p \overset{e}{\Longrightarrow} ?j \overset{z}{\mapsto} r_j}$$

E3

$$\frac{p \overset{f}{\Longrightarrow} \Omega}{p \overset{e}{\Longrightarrow} \Omega}$$

Lemme 6 *Soient p' et q' des preuves partielles et sans coupures respectivement du séquent $\Gamma_1, \Delta; \wedge_i A_i$ et du séquent $\Gamma_2, \Delta, \vee_i A_i^\perp$, la formule $\vee_i A_i^\perp$ étant de nom x. Alors, pour toutes preuves partielles p et q telles que $p \overset{e}{\Longrightarrow} p'$ et $q \overset{e}{\Longrightarrow} q'$, il existe une preuve partielle et sans coupure r vérifiant* $\text{Coupe}_x(p; q) \overset{e}{\Longrightarrow} r$.

PREUVE : Par induction structurelle sur $\wedge_i A_i$, puis, par induction structurelle sur q', et enfin, par induction sur la preuve de $q \stackrel{e}{\Longrightarrow} q'$.

Soient donc p et q des preuves partielles, p' et q' des preuves partielles et sans coupures respectivement de $\Gamma_1, \Delta; \wedge_i A_i$ et de $\Gamma_2, \Delta, \vee_i A_i^\perp$, telles que $p \stackrel{e}{\Longrightarrow} p'$ et $q \stackrel{e}{\Longrightarrow} q'$.

Du fait que $\wedge_i A_i$ est conjonctif, p et p' s'écrivent respectivement $?i \stackrel{y}{\mapsto} p_i$ et $?i \stackrel{y}{\mapsto} p'_i$ et, de par la définition de $\stackrel{e}{\Longrightarrow}$, on a $p_i \stackrel{e}{\Longrightarrow} p'_i$.

La preuve q' est sans coupure et prouve un séquent sans formule conjonctive. Ou bien q' est Ω ou bien elle a la forme $!(x', i'_0)q'_0$ avec $q'_0 = ?j \stackrel{z}{\mapsto} q'_j$.

Dans le premier cas, on en déduit que $q \stackrel{f}{\Longrightarrow} \Omega$, puis $\mathrm{Coupe}_x(p;q) \stackrel{d}{\Longrightarrow} \Omega$, puis $\mathrm{Coupe}_x(p;q) \stackrel{f}{\Longrightarrow} \Omega$ et, enfin, $\mathrm{Coupe}_x(p;q) \stackrel{e}{\Longrightarrow} \Omega$, ce qui est le résultat attendu.

Dans le second cas, de par la définition de $\stackrel{e}{\Longrightarrow}$, il existe q_0 telle que $q \stackrel{f}{\Longrightarrow} !(x, i_0)q_0$ et $q_0 \stackrel{e}{\Longrightarrow} q'_0$.

De par la définition de $\stackrel{f}{\Longrightarrow}$, plusieurs cas se présentent.

- S'il existe q'' tel que $q \stackrel{f}{\Longrightarrow}_1 q''$ et $q'' \stackrel{f}{\Longrightarrow} !(x, i_0)q_0$ alors, on a aussi $q'' \stackrel{e}{\Longrightarrow} q'$ mais de manière plus courte que pour avoir $q \stackrel{e}{\Longrightarrow} q'$. Il s'ensuit, par hypothèse d'induction, qu'il existe r tel que $\mathrm{Coupe}_x(p;q'') \stackrel{e}{\Longrightarrow} r$. La preuve sans coupure r doit avoir une forme $!(t, k_0)r_0$ et il doit exister r'_0 telle que $\mathrm{Coupe}_x(p;q'') \stackrel{f}{\Longrightarrow} !(t, k_0)r'_0$ et $r'_0 \stackrel{e}{\Longrightarrow} r_0$. Mais $q \stackrel{f}{\Longrightarrow}_1 q''$ entraine aussi
$$\mathrm{Coupe}_x(p;q) \stackrel{f}{\Longrightarrow}_1 \mathrm{Coupe}_x(p;q'')$$
d'où $\mathrm{Coupe}_x(p;q) \stackrel{f}{\Longrightarrow} !(t, k_0)r'_0$ puis $\mathrm{Coupe}_x(p;q) \stackrel{e}{\Longrightarrow} r$.
- Si $q \stackrel{d}{\Longrightarrow} !(x', i'_0)q_0$ et que $x \neq x'$ alors, posant $q_0 = ?j \stackrel{z}{\mapsto} q_j$, on a
$$\mathrm{Coupe}_x(p;q) \stackrel{d}{\Longrightarrow} !(x', i'_0)?j \stackrel{z}{\mapsto} \mathrm{Coupe}_x(p;q_j)$$
et donc $\mathrm{Coupe}_x(p;q) \stackrel{f}{\Longrightarrow} !(x', i'_0)?j \stackrel{z}{\mapsto} \mathrm{Coupe}_x(p;q_j)$. Mais $q_0 \stackrel{e}{\Longrightarrow} q'_0$ donc, pour chaque j, $q_j \stackrel{e}{\Longrightarrow} q'_j$ et les q'_j sont plus petits que les q_j. Donc, par hypothèse d'induction, on obtient l'existence de r_j telle que $\mathrm{Coupe}_x(p;q_j) \stackrel{e}{\Longrightarrow} r_j$. Il suffit alors de poser $r = !(x', i'_0)?j \stackrel{z}{\mapsto} r_j$ pour pouvoir affirmer que $\mathrm{Coupe}_x(p;q) \stackrel{e}{\Longrightarrow} r$.
- Si $q \stackrel{d}{\Longrightarrow} !(x', i'_0)q_0$ avec $x = x'$ et que $q_0 = ?j \stackrel{z}{\mapsto} q_j$ alors on a $\mathrm{Coupe}_y(p;q) \stackrel{f}{\Longrightarrow}_1 \mathrm{Coupe}_x(?j \stackrel{z}{\mapsto} \mathrm{Coupe}_x(p;q_j); p_{i_0})$. Mais, puisque les q_j sont plus petits que q et que $q_0 \stackrel{e}{\Longrightarrow} q'_0$ entraine $q_j \stackrel{e}{\Longrightarrow} q'_j$ avec q'_j sans coupure, on obtient, par hypothèse d'induction, l'existence de r_j tel que $\mathrm{Coupe}_x(p;q_j) \stackrel{e}{\Longrightarrow} r_j$.

  En appliquant l'hypothèse d'induction pour $?j \stackrel{z}{\mapsto} \mathrm{Coupe}_z(p;q_j)$ et $?j \stackrel{z}{\mapsto} r_j$, et pour p_{i_0} et p'_{i_0}, puisque la formule de coupure, maintenant A_{i_0}, est plus petite, on obtient l'existence de r tel que $\mathrm{Coupe}_y(?j \stackrel{z}{\mapsto} \mathrm{Coupe}_z(p;q_j); p_{i_0}) \stackrel{e}{\Longrightarrow} r$.

■

Un corollaire immédiat s'obtient en prenant pour p et q des preuves sans coupure.

Proposition 7 *Si p et q sont des preuves partielles et sans coupure respectivement de $\Gamma_1, \Delta; \wedge_i A_i$ et de $\Gamma_2, \Delta, \vee_i A_i^\perp$, alors, il existe r partiel et sans coupure tel que* $\mathrm{Coupe}(p;q) \stackrel{e}{\Longrightarrow} r$.

Corollaire 2 *Si p et q sont des preuves partielles et sans coupures respectivement de $\Gamma_1, \Delta; \wedge_i A_i$ et de $\Gamma_2, \Delta, \vee_i A_i^\perp$, alors, il existe r sans coupure en tête tel que* $\mathrm{Coupe}(p;q) \stackrel{f}{\Longrightarrow} r$. *Autrement dit l'élimination faible de tête de* $\mathrm{Coupe}(p;q)$ *termine en un nombre d'étapes fini.*

PREUVE : De par la définition de $\stackrel{e}{\Longrightarrow}$. ■

On peut aussi étendre le résultat à des preuves avec coupure.

Proposition 8 *Pour toute preuve partielle p, il existe une preuve partielle r sans coupure telle que $p \stackrel{e}{\Longrightarrow} r$.*

PREUVE : Par induction sur p.

Si p a la forme $?j \stackrel{z}{\mapsto} p_j$ ou $!(x, j_0)p_0$, on obtient le résultat par l'hypothèse d'induction.

Si p a la forme $\mathrm{Coupe}(p_1; p_2)$ alors, par hypothèse d'induction, on a l'existence de r_1 et r_2 sans coupure tels que $p_1 \stackrel{e}{\Longrightarrow} r_1$ et $p_2 \stackrel{e}{\Longrightarrow} r_2$. On peut donc appliquer le lemme avec r_1, r_2, p_1 et p_2, ce qui fournit r sans coupure tel que $p \stackrel{e}{\Longrightarrow} r$. ■

Corollaire 3 *Si p est une preuve partielle alors la réduction faible de tête de p termine en un nombre fini d'étapes.*

Corollaire 4 *Si p est une preuve partielle du séquent vide alors $p \stackrel{f}{\Longrightarrow} \Omega$.*

Remarques : 1) Le lemme-clé de ces preuves est à mettre à l'actif de Coquand.

2) Ces résultats, à l'exception du corollaire 4, restent vrais si l'on considère des preuves non partielles.

3) Ces résultats sont aussi transposables à d'autres procédures d'élimination des coupures. Par exemple en autorisant des contractions non seulement sur les formules disjonctives mais aussi sur les formules conjonctives, et en appliquant, par exemple, la technique dite des coupures croisées.

4) Dans un cadre sans règle infinitaire, on peut utiliser le résultat de Gonthier, Lévy et Méllies [28] pour inférer à partir de la preuve d'existence d'une forme normale (par exemple en éliminant les coupures selon une stratégie de réduction de la coupure la plus interne d'abord) l'existence d'une réduction standard normalisante, permettant de dire que l'élimination des coupures selon une stratégie de réduction de la coupure la plus externe d'abord termine.

6.5 L'équivalence

On se propose de montrer, étant donnée deux preuves partielles de Novikoff prouvant des formules duales l'une de l'autre, que la réduction faible d'une coupure entre ces preuves suit les mêmes étapes que le débat entre ces preuves vues comme stratégies partielles non-perdantes. En particulier, la réduction faible termine si et seulement si le débat termine.

Les preuves de Novikoff n'étant pas stables par élimination des coupures en raison de redondances pouvant apparaître lors de la réduction, on les considère comme des preuves "standard", i.e. avec d'éventuelles redondances, et on applique les résultats des sections précédentes sur l'élimination des coupures. [1].

Une conséquence directe du corollaire 4 est le résultat suivant :

Proposition 9 *Soient p et q des preuves partielles de Novikoff, respectivement de $;\wedge_i A_i$ et de $\vee_i A_i^\perp$. Alors la réduction faible de tête de $\mathrm{Coupe}(p; q)$ débouche en un nombre fini d'étape sur Ω.*

Une autre manière de réduire la coupure entre p et q et d'appliquer la procédure définie par Coquand sous le nom de débat qui consiste à laisser jouer l'une contre l'autre p et q vues comme des stratégies partielles non-perdantes. Nous montrons que les deux approches partagent les mêmes étapes de calcul.

[1] Une autre solution aurait été d'introduire une règle de contraction des redondances (si Γ, A, A alors Γ, A) appliquable à chaque étape de création potentielle d'une redondance (i.e. dans la règle de réduction R1). Cette règle, comme la règle de coupure aurait eu un aspect calculatoire et ses propres règles de réduction. Le monde des preuves de Novikoff aurait alors été stable par élimination des coupures *et* des contractions. Cette approche est celle suivie, par exemple, par Novikoff [46]. Aurait-elle évité le recours à l'outil des noms ?

6.5.1 Preuves bien agencées

On étudie ici la forme bien particulière que les preuves apparaissant dans la trace d'une réduction faible ont.

Soit p et q des preuves partielles sans coupures respectivement de $\Gamma_1, \Delta; \wedge_i A_i$ et de $\Gamma_2, \Delta, \vee_i A_i^\perp;$.

Lorsqu'on regarde les étapes successives de la réduction faible de tête de Coupe$(p;q)$, il s'avère que toutes les preuves impliquées dans la trace ont une structure non innocente. On les qualifiera naïvement de preuves "bien agencées".

- une preuve partielle sans coupure est une **preuve bien agencée**.
- si $(p_j)_j$ est une famille de preuves bien agencées et si q est une preuve bien agencée, alors Coupe$(?j \mapsto p_j; q)$ est une **preuve bien agencée**

Ainsi, une preuve bien agencée peut être vue comme un arbre dont les nœuds sont annotés par des règles de coupure et les feuilles par des preuves partielles sans coupures.

Nous associons des occurrences aux nœuds et feuilles de l'arbre. L'**occurrence** de la racine de l'arbre est notée ϵ. Si un sous-arbre Coupe$(?j \mapsto p_j; q)$ a l'occurrence w alors p_j a l'occurrence wj et q a l'occurrence $w\alpha$, où α est un symbole spécial introduit pour l'occasion. En particulier, on n'associe pas d'occurrence au sous-arbre direct gauche $?j \mapsto p_j$ de la coupure mais seulement à ses propres sous-arbres p_j.

On ordonne les occurrences de nœuds et feuilles par un ordre lexicographique noté $\prec$. Au niveau des "lettres" de l'"alphabet", on pose $i \prec_L \alpha$ quelque soit l'indice i et on n'ordonne pas les différents i entre eux. L'ordre $\prec$ est ainsi partiel. On dit aussi que w est un préfixe de w' si w s'écrit $w'w''$ pour une certaine suite de "lettres" w''.

A tout nœud d'occurrence w dans p, on associe de manière canonique une feuille $F_w(p)$, seule feuille dont l'occurrence a la forme $w\alpha^n$. Autrement dit, la feuille d'occurrence $F_w(p)$ est la feuille d'extrême-droite du sous-arbre situé au nœud d'occurrence w. En particulier, la feuille $F_\epsilon(p)$ est la feuille la plus à droite de p. Réciproquement, toute feuille s'écrit de manière unique sous la forme $F_w(p)$ pour une certaine occurrence w ne se terminant par α.

On notera $|F_w(p)|$ le séquent prouvé par la preuve sans coupure située en la feuille d'occurrence $F_w(p)$. On notera $x \in |F_w(p)|$ pour dire que x est un nom utilisé dans ce séquent. Si l'occurrence associée à x fait référence à la sous-formule $\vee_{j\in J} B_j$ alors (x, i) pour $i \in I$ est un coup possible pour la preuve située en $F_w(p)$ et on utilisera la même notation $x, i_0 \in |F_w(p)|$ pour l'exprimer.

Si $x \in |F_w(p)|$, on peut suivre la trace de x en remontant vers la racine, quitte à ce que x change de nom en traversant les coupures (de par les fonctions de renommage). Il peut arriver trois sortes de destinées à ce nom x. Ou bien il traverse l'arbre et apparaît dans le séquent prouvé par la racine de l'arbre, ou bien il est **capturé** par l'une des coupures rencontrées, ou alors, il est **gelé** par une des coupures.

On dit qu'un nom $x \in |F_w(p)|$ **apparaît** sous le nom x' à l'occurrence w' dans les cas suivants :

- $w' = F_w(p)$ et $x = x'$
- si x apparaît sous le nom x' à l'occurrence $w'\alpha$ tandis que le nœud w' a la forme Coupe$_{(z,Y,\sigma)}(?i \overset{y}{\mapsto} p_i; q)$ avec $z \neq x'$, alors x apparaît sous le nom $\sigma(x')$ à l'occurrence w'
- si x apparaît sous le nom x' à l'occurrence $w'i$ alors il apparaît sous le nom x' à l'occurrence w'

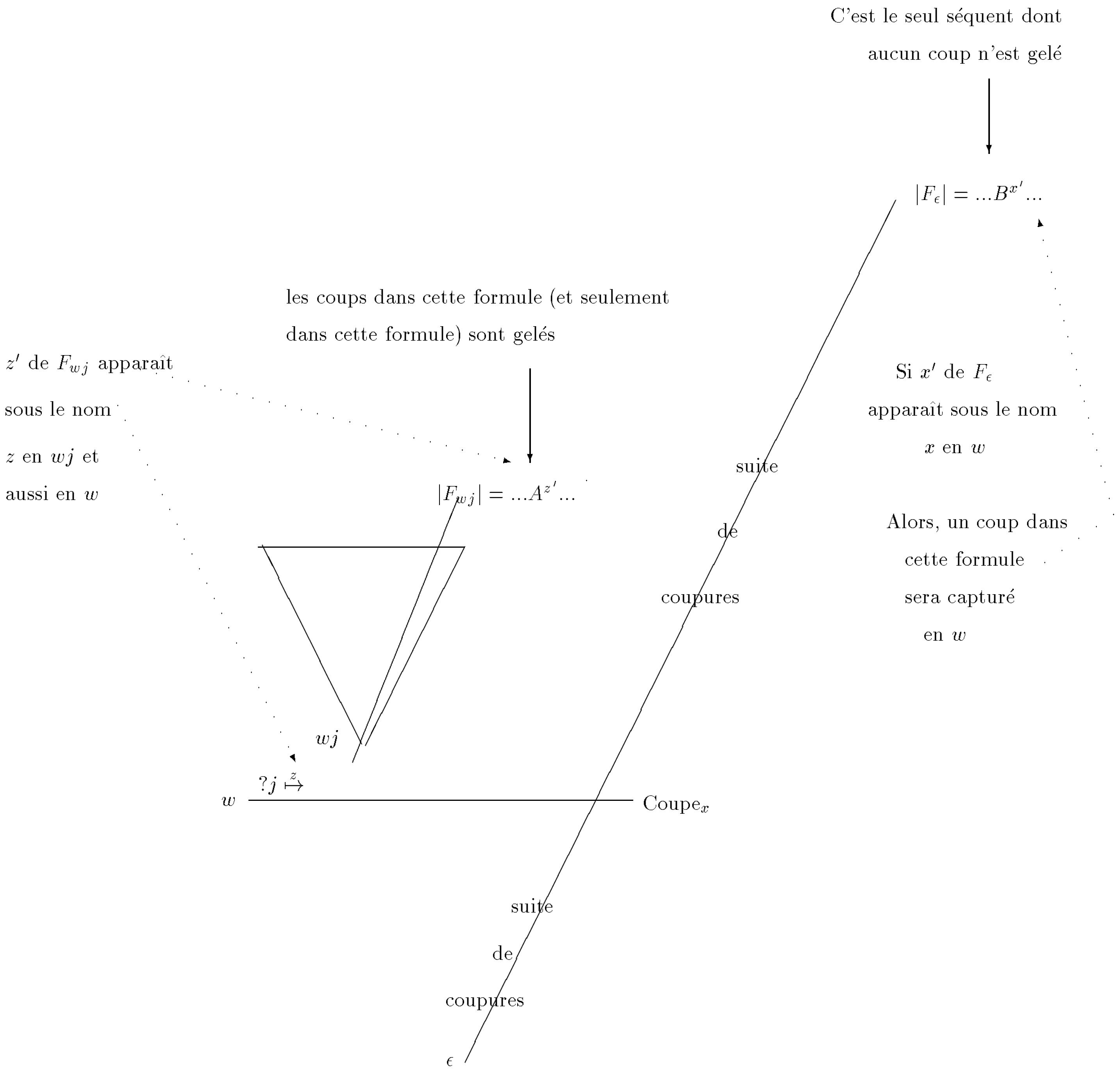
On dit qu'un $\vee$-coup $x, i_0 \in |F_w(p)|$ est **capturé** à l'occurrence w' si on a les conditions suivantes :

- x apparaît sous le nom x' à l'occurrence $w'\alpha$
- w' désigne une coupure Coupe$_{x'}(p;q)$

Remarque : En ce cas, w' est un préfixe de w.

On dit qu'un nom $y \in |F_{wi}(p)|$ ou qu'un coup $y, j_0 \in |F_{wi}(p)|$ est **gelé** si le nœud d'occurrence w a la forme Coupe$_x(?i \overset{y'}{\mapsto} p_i; q)$ et si y apparaît sous le nom y' en wi.

Voici un résumé schématique des notions sur les preuves agencées :



Remarque : Si p est une preuve bien agencée et que $p \overset{f}{\Longrightarrow}_1 q$ alors q est construite à partir de "morceaux" de p de telle sorte que q est bien agencée. On définit une fonction de projection des nœuds et feuilles de q vers les nœuds et feuilles de p.

Soit p une preuve bien agencée telle que $F_\epsilon(p) = !(x, i_0)?j \mapsto q_j$, que x, i_0 soit capturé en α^n et que $p \overset{f}{\Longrightarrow}_1 q$. A toute occurrence de nœud ou de feuille de q, à l'exception de α^n et des $F_{\alpha^n j}(q)$, on associe une occurrence dans p :

1. $\mathcal{T}(\alpha^m w) = \alpha^m w$ pour $m < n$ et w ne commençant pas par α

2. $\mathcal{T}(\alpha^n jw) = \alpha^n w$
3. $\mathcal{T}(\alpha^{n+1}w) = \alpha^n i_0 w$

Remarque : Si w a une forme telle que $F_w(q)$ soit l'occurrence d'une feuille alors $F_{\mathcal{T}(w)}(p)$ aussi.

Lemme 7 *Si $F_\epsilon(p) =!(x, i_0)?j \mapsto q_j$, que x, i_0 est capturé en α^n et que $p \stackrel{f}{\Longrightarrow}_1 q$, alors, on a*

- $F_w(q) = F_{\mathcal{T}(w)}(p)$ *si* $w \neq \alpha^n j$
- $F_{\alpha^n j}(q) = q_j$

PREUVE : Par construction de q à partir de p. ∎

Soit $F_w(p)$ une feuille de p. Tout coup $x, i_0 \in |F_w(p)|$ est un **coup jouable** de $F_w(p)$ à la condition qu'il ne soit pas gelé. On note $\mathcal{J}_w(p)$ l'ensemble des coups jouables de $F_w(p)$.

Lemme 8 *Soit p une preuve bien agencée telle que $F_\epsilon(p) =!(x, i_0)?j \stackrel{z}{\mapsto} q_j$, que x, i_0 soit capturé en α^n et que $p \stackrel{f}{\Longrightarrow}_1 q$. Soit w une occurrence dans q telle que $F_w(q)$ ait un sens.*

1. *Si $x \in |F_{\mathcal{T}(w)}(p)|$ apparaît sous le nom x' en w' alors $x \in |F_w(q)|$ et il existe w'' tel que $x \in |F_w(q)|$ apparaît sous le nom x' en w'' avec $w' = \mathcal{T}(w'')$.*
2. *Si $x \in |F_{\mathcal{T}(w)}(p)|$ est capturé en w' alors x est un nom de $|F_w(q)|$ et il existe w'' tel que $x \in |F_w(q)|$ soit capturé en w'' avec $w' = \mathcal{T}(w'')$.*
3. *Si $x \in |F_{\mathcal{T}(wk)}(p)|$ est gelé alors x est un nom gelé de $|F_w k(q)|$.*
4. *Pour $w \neq \epsilon$ et $w \neq \alpha^n j$, si $t, k_0 \in |F_w(q)|$ alors $t, k_0 \in |F_{\mathcal{T}(w)}(p)|$.*
5. *Pour $w \neq \epsilon$ et $w \neq \alpha^n j$, si $c \in \mathcal{J}_w(q)$ alors $c \in \mathcal{J}_{\mathcal{T}(w)}(p)$. Si $c \in \mathcal{J}_{\alpha^n j}(q)$ alors $c \in \mathcal{J}_\epsilon(p)$.*
6. *Si, de plus, c, vu comme coup jouable dans p, est capturé en w', alors, il existe w'' tel que $T(w'') = w'$ et c, vu cette fois comme joup jouable dans q, est capturé en w''.*

PREUVE : Le 1 s'obtient par induction sur la preuve d'apparition, tenant compte de la manière de construire q à partir de p et en particulier de la brisure de linéarité de $\mathcal{T}$ en $\alpha^n j$ et α^{n+1}. Le 2 et le 3 s'ensuivent directement du 1. Le 4 s'obtient à partir du lemme 7.

Pour le 5, si $c \in \mathcal{J}_w(q)$ alors $c \in |F_w(q)|$ donc $c \in |F_{\mathcal{T}(w)}(p)|$. Il n'est pas gelé dans $F_w(q)$ donc, si $w \neq \alpha^n j$, pas gelé dans $F_{\mathcal{T}(w)}(p)$ par le 3, et donc jouable. Si $w = \alpha^n j$ alors, par construction, z est gelé et les coups de la forme z, k_0 ne sont pas jouables. Autrement dit les coups jouables de $F_{\alpha^n j}(q)$ sont dans $|F_\epsilon(p)|$. Mais comme aucun des coups de ce dernier n'est gelé, les coups jouables de $F_{\alpha^n j}(q)$ sont jouables dans F_ϵ.

Le 6 vient du 2. ∎

6.5.2 Conformité

Dans la suite, par preuve de la trace, on sous-entend une preuve agencée de la trace de la réduction faible de la coupure entre deux preuves données de Novikoff.

Soit p une preuve de la trace et $F_w(p)$ une feuille de cette preuve. On note $P_w(p)$ la position associée à la feuille $F_w(p)$.

Soit p une preuve de la trace et $F_w(p)$ une feuille de cette preuve. Soit $\pi = (f_0, u_0), ..., (f_n, u_n)$ une partie finie de position courante $l_n = u_{m_1}...u_{m_n}$.

On dit que p est **conforme** à π en w si les conditions suivantes sont réunies :

- $P_w(p) = l_n$

- pour tout $x, i_0 \in \mathcal{J}_w(p)$, tel que x désigne u_{m_k}, on a x capturé en un nœud d'occurrence w' et, en plus, p conforme à $\pi|_{m_k-1}, (m_k, u_{m_k} i_0)$ en $w' i_0$ (par convention, si $m_k = 0$, $\pi|_{m_k-1}$ désigne la partie vide)

Si p est conforme à π en la racine, on dit simplement que p est **conforme** à π.

Lemme 9 *Soit r_n un élément de la trace $r_0, ..., r_n, r$ de la réduction faible de* Coupe$(p; q)$. *Alors, interprétant p et q comme des stratégies partielles non perdantes, si n est pair on a $q(P_\epsilon(r_n)) = C_\epsilon(r_n)$, et si n est impair, on a $p(P_\epsilon(r_n)) = C_\epsilon(r_n)$.*

PREUVE : De par la construction de l'isomorphisme entre preuves de Novikoff et stratégies partielles non perdantes. ■

Pour les lemmes suivants, p est une preuve de la trace telle que $F_\epsilon(p)$ ait la forme $!(x, i_0)?j \mapsto q_j$, que $C(F_\epsilon(p)) = x, i_0$ soit capturé en un nœud α^n et telle que $p \overset{f}{\Longrightarrow}_1 q$. La suite $\pi = (f_0, u_0), ..., (f_n, u_n)$ est une partie finie et $F_w(q)$ une feuille dans q telle que p soit conforme en $\mathcal{T}(w)$ à π et que x fasse référence à u_m.

Lemme 10 *Si $F_w(q) \prec \alpha^n$ alors q est conforme à π en w.*

PREUVE : Par induction sur la preuve de conformité de p en $\mathcal{T}(w)$. Puisque, par le lemme 7, $F_w(q) = F_{\mathcal{T}(w)}(p)$, on a, par conformité de p en $\mathcal{T}(w)$, $F_w(q) = l_n$. Soit maintenant $z, k_0 \in \mathcal{J}_w(q)$ tel que z fasse référence à u_f. Par conformité de p, ce coup, vu comme coup de $\mathcal{J}_{\mathcal{T}(w)}(p)$ est capturé en une occurrence w'. Par le lemme 8, il existe w'' tel que c, vu comme coup de $\mathcal{J}_w(q)$ soit capturé en w'' avec $\mathcal{T}(w'') = w'$. Or, par définition de la conformité, p est conforme en $w' k_0$ à $\pi|_{f-1}, (f, u_f k_0)$.

Puisque w'' est un préfixe de w, on a $F_{w'' k_0}(q) \prec \alpha^n$. On peut donc appliquer l'hypothèse de récurrence pour conclure que q est conforme à $\pi|_{f-1}, (f, u_f k_0)$ en $w'' k_0$.

On en déduit que q est conforme à π en w. ■

Lemme 11 *Si $\alpha^n ji \prec F_w(q) \preceq F_{\alpha^n ji}(q)$ pour un certain i et un certain j, alors q est conforme à π aussi en w.*

PREUVE : La preuve est identique sauf que maintenant, soit $F_{w'' k_0}(q) \prec \alpha^n$ auquel cas le lemme précédent s'applique pour donner que q est conforme à $\pi|_{f-1}, (f, u_f k_0)$ en $w'' k_0$, soit $\alpha^n ji \prec F_{w'' k_0}(q)$ auquel cas, par induction, on a aussi q conforme à $\pi|_{f-1}, (f, u_f k_0)$ en $w'' k_0$.

On en déduit que q est conforme à π en w. ■

Lemme 12 *Si $F_{\alpha^n ji}(q) \prec F_w(q) \preceq F_{\alpha^n j}$ pour tout i et un certain j, alors q est conforme à π en w.*

PREUVE : La preuve est aussi identique sauf que maintenant, ou bien $F_{w'' k_0}(q) \prec \alpha^n$ auquel cas le lemme 10 s'applique, ou bien $w'' = \alpha^n j k_0$ (i.e. "k" est un "i"), auquel cas le lemme 11 s'applique, ou alors $F_{\alpha^n ji}(q) \prec F_{w'' k_0}(q) \preceq F_{\alpha^n j}$ auquel cas, par induction, q est conforme à $\pi|_{f-1}, (f, u_f k_0)$ en $w'' k_0$.

On en déduit que q est conforme à π en w. ■

Lemme 13 *Si $w = \alpha^n j$ pour un certain j, alors q est conforme à $\pi, (m, u_m)$ en w.*

PREUVE : La preuve est identique à celle du lemme 12 sauf qu'on n'a pas $F_w(q) = F_{\mathcal{T}(w)}(p)$ mais $F_w(q) = q_j$. Mais la position de q_j est précisément $\pi, (m, u_m)$ d'où le résultat. ■

Lemme 14 *Si $F_{\alpha^n j}(q) \prec F_w(q) \prec F_\epsilon$ pour tout j, alors q est conforme à π en w.*

PREUVE : Maintenant, ou bien $F_{w'' k_0}(q) \prec \alpha^n$ auquel cas le lemme 10 s'applique, ou bien $F_{\alpha^n j}(q) \prec F_{w'' k_0}(q) \prec F_\epsilon$ auquel cas, par induction, q est conforme à $\pi|_{f-1}, (f, u_f k_0)$ en $w'' k_0$.

Le cas $w'' = \alpha^n$ n'est pas possible car $\mathcal{T}(w'') = w'$ et w'' n'est pas dans le domaine de définition de $\mathcal{T}$. On en déduit que q est conforme à π en w. ■

Proposition 10 *Soit p une preuve de la trace conforme à une partie $\pi = (f_0, u_0), ..., (f_n, u_n)$. On suppose que $C(F_\epsilon(p)) = (x, i_0)$, que x désigne u_f et que $p \overset{f}{\Longrightarrow}_1 q$. Alors q est conforme à $\pi, (f, u_f i_0)$.*

PREUVE : On désigne par w le nœud en lequel le coup x, i_0 est capturé. Par la conformité de p, on sait que $P_{wi_0}(p) = l_{f-1}(u_f i_0)$. Or $P_\epsilon(q) = P_{wi_0}(p)$ tandis que $l_{n+1} = l_{f-1}(u_f i_0)$. On a donc $P_\epsilon(q) = l_{n+1}$.

Soit maintenant z, k_0 un coup jouable de $F_\epsilon(q)$. Il y a deux cas à considérer.

La première hypothèse est que z, k_0 était déjà jouable dans $P_{wi_0}(p)$, c'est-à-dire que z ne désigne pas $u_f i_0$ mais un certain u_m. En ce cas, par conformité de p, ce nom z est capturé en un nœud w'. Par le lemme 8, il existe donc w'' tel que $\mathcal{T}(w'') = w'$ et $z \in |F_\epsilon(q)|$ est capturé en w''. Puisque w'' est un préfixe de $F_\epsilon(q)$, il est constitué uniquement de α et donc, $w''k_0$ rentre dans l'une des catégorie des lemmes 10 ou 13. On a donc la conformité de q à $\pi|_{m-1}, (m, u_m)$ en $w''k_0$.

Dans le second cas, z est le nom de $u_f i_0$ et était gelé dans $P_{wi_0}(p)$. C'est donc w qui capture le coup z, k_0. D'autre part ce $\vee$-coup correspond au J-coup $(n+1, u_f i_0 k_0)$ et il s'agit de montrer que q est conforme à $\pi, (n+1, u_f i_0 k_0)$ en wk_0. Or, par le lemme 13 (le "k_0" est en fait un j), on obtient que q est conforme à π en wk_0.

Pour tous les coups jouables, on obtient la conformité de q en l'endroit de capture. On en déduit la conformité de q à π. ■

Théorème 3 (Correspondance entre réduction faible et débat)
Soient p et q des preuves partielles de Novikoff respectivement de $;\wedge_i A_i$ et de $\vee_i A_i^\perp$. Si $r_0, ..., r_n, r$ est la trace de $\mathrm{Coupe}(p; q) \overset{f}{\Longrightarrow} r$ *et si $(f_0, u_0), ..., (f_{n'}, u_{n'})$ est le débat entre p et q, vus comme des stratégies partielles non perdantes, alors, pour tout $m \leq n$, $C(F_\epsilon(r_m))$ est défini si et seulement si u_m est défini et, lorsque définis, $\mathcal{O}cc(C(F_\epsilon(r_m))) = u_m$. En particulier $n = n'$.*

PREUVE : Parce que $\mathrm{Coupe}(p; q)$ est conforme à <> et par la proposition 10. ■

Remarque : On obtient par ce résultat, conjointement au résultat de terminaison de l'élimination faible de tête des coupures, une preuve de terminaison du débat. Cette preuve diffère de la première preuve de Coquand (dans la première version de [9]) en ce qu'elle est constructive et repose sur la bonne fondaison des formules. La preuve de Coquand, en revanche, utilise un argument de logique classique mais fonctionne aussi avec des formules non bien fondées.

Conclusion

La correspondance de Curry-Howard mettait en relation la logique combinatoire avec les systèmes logiques de type axiomatique ainsi que le λ-calcul avec la déduction naturelle. Nous pensons, avec le $\overline{\lambda}$-calcul et LJT, avoir l'extension directe de cette correspondance dans le cadre de la structure de calcul de séquents (l'adaptation au cas classique via le $\overline{\lambda}$-calcul et le calcul LKT ne posant pas de problèmes). De plus, nous obtenons avec le $\overline{\lambda}$-calcul un raffinement strict du λ-calcul usuel au même titre que le λ-calcul est un raffinement strict de la logique combinatoire : tout terme du λ-calcul peut se coder dans le $\overline{\lambda}$-calcul de telle manière que toute β-réduction se traduise par une suite de réductions dans le $\overline{\lambda}$-calcul. Cependant, pour affirmer cela, il faut considérer le système de réduction enrichi décrit dans la remarque de la section 2.7, système pour lequel la terminaison forte reste une conjecture.

En dépit de ce résultat sur LJT et LKT, nous ne prétendons pas que le calcul des séquents doive se glisser dans le même moule fonctionnel que la déduction naturelle et ainsi se limiter aux calculs LJT et LKT. Au contraire, la constatation des nombreuses versions de "calcul des séquents" suggère que celui-ci réfère plus à une structure de système logique caractérisée par la présence uniquement de règles d'introduction qu'à un système bien particulier.

Et autant l'extension de la correspondance de Curry-Howard au calcul des séquents semble imposer de considérer LJT et LKT, autant l'interprétation au travers d'un regard "théorie des jeux" semble être propre à la structure de calcul des séquents et non à un calcul bien particulier.

Même si nous n'avons étudié que les exemples de LJQ et LKQ, pour leur correspondance avec les E-dialogues de Lorenzen, et du calcul de Novikoff pour l'intérêt que Coquand lui a porté, cette interprétation reste faisable dans la plupart des variantes de calcul des séquents. Nous imaginons aussi pouvoir transcrire l'analogue du débat en calcul de Novikoff dans n'importe quel calcul des séquents adéquat.

Il est remarquable aussi que les jeux associés aux calculs des séquents sont plus simples lorsqu'on considère le cas classique que lorsqu'on se restreint aux cas minimal ou intuitionniste.

La question se pose aussi de savoir s'il suffit de considérer des jeux asymétriques de type E-dialogue ou bien s'il vaut la peine d'introduire des notions symétriques de jeux, typiquement des jeux dont le débat constituerait une partie. Une telle approche est par exemple prise par Abramsky *et al* [1] et Hyland *et al* [33].

Cette thèse, à la fois, apporte des réponses avec LJT et LKT, et des interrogations avec les interprétations en termes de jeux.

Le calcul des séquents avait jusqu'alors une place ambiguë en informatique. Utilisé dans le domaine de la recherche automatique de preuves en raison de ses propriétés métamathématiques (propriété de la sous-formule), il était peu utilisé pour la formalisation du raisonnement, n'ayant ni un aspect calculatoire clair, ni l'aspect "naturel" de la déduction naturelle.

Ce qu'apporte cette thèse, c'est que LJT et LKT, tout en bénéficiant des propriétés métamathématiques du calcul des séquents, intègrent aussi les propriétés calculatoires de la déduction naturelle. De plus, les λ-calculs associés à LJT et LKT sont proches des machines à environnement (présence d'une opération explicite de substitution, analogie entre liste d'arguments et pile d'arguments). Ne sont-ce pas des caractéristiques qui destinent LJT ou LKT a être le formalisme interne des implémentations de systèmes formels ?

Quelle place faut-il accorder à l'approche de la prouvabilité en termes de jeux ? Y a-t-il plus qu'une interprétation calculatoire de telle ou telle version de calcul des séquents par telle ou telle notion de jeux ad hoc ? Les jeux apportent-ils une intuition nouvelle sur la prouvabilité ?

Peut-on espérer, se réfèrant à une étude comme celle de Moschovakis [44] qui recourt aux jeux pour modéliser des programmes interactifs, une extension de l'analogie entre preuves et programmes à des objets informatiques plus complexes tels les programmes interactifs ?

Annexe

Elimination des coupures en calcul de Novikoff

Nous donnons un exemple de réalisation en langage ML (précisément en Caml-Light) des procédures de réduction faible de tête des coupures et de la procédure d'élimination des coupures suivant la stratégie de réduction de la coupure la plus externe d'abord.

La représentation des preuves/termes

On prend comme ensemble de noms, l'ensemble des objets ML de type `string`.

```
type nom == string;;
```

On construit, à partir du nom x, un nouveau nom `F(X,x)` n'appartenant pas à `X` en considérant le nom `"x"` et, le cas échéant, `"x'"`, etc...

```
let rec est_présent x = function
   [] -> false
 | y::l -> x=y or (est_présent x l);;

let F (noms,x) = essaie (x^"#")
  where rec essaie x = if (est_présent x noms) then essaie (x^"#") else x;;
```

On considère les fonctions de renommages comme des listes de couples *(antécédent,image)*.

```
type renommage == (nom * nom) list;; (* fonction de renommage à domaine fini *)

let sigma_vide = [];;

let ajoute (x',x) sigma = (x',x)::sigma;;

let associe x sigma = cherche sigma
  where rec cherche = function
    []        -> x
  | (y,y')::l -> if x=y then y' else cherche l;;
```

On appelle annotations les informations servant à la gestion des noms dans la règle de coupure.

```
type annotation == nom * nom list * renommage;;
```

A défaut d'avoir, en ML, un typage d'ordre supérieur, on admet que toutes les disjonctions et conjonctions partagent le même ensemble `index` d'indexation. Cependant la structure des éléments de `index` exprimera la nature de l'ensemble d'indexation. Par exemple, les membres d'une disjonction ou d'une conjonction binaire seront indexés par les index `B true` et `B false`. Ceux d'une disjonction ou d'une conjonction ternaire par `D Gauche`, `D Milieu` et `D Droite`. Les membres d'une quantification sur un entier seront indexés par les index de la forme `E n` où n est un entier. Les membres d'une quantification sur un triplet (le cas arrive dans l'exemple ci-dessous) seront indexés par des index de la forme `T` ... , etc.

D'autre part, un atome vrai, membre d'une formule conjonctive, sera considéré comme la disjonction unaire de la formule vraie (ceci afin de respecter l'alternance disjonctif/conjonctif). C'est le rôle de l'index `S` (`S` comme singleton).

Enfin, l'index `O` (comme "ensemble vide") est là pour définir la proposition "**1**". Cet index ne devra jamais apparaître dans une règle d'introduction du $\vee$ (car il exprimerait alors jouer dans l'atome "**0**").

```
type ternaire = Gauche | Milieu | Droite;;

type index =
   E of int
 | B of bool
 | D of ternaire
 | T of index * index * index
 | S
 | O;;

type preuve =
   OU_INTRO of nom * index * preuve
 | ET_INTRO of index -> (nom * preuve)
 | COUPE of annotation * preuve * preuve
 | INDETERMINE;;
```

Remarque : Parmi les objets de type `preuve`, auront seuls du sens ceux correspondant à des preuves vérifiant les contraintes — non explicites — imposées par les séquents.

La réduction faible de tête

```
let rec faible p =
  match p with
   COUPE((x,X,sigma), p, q) ->
   begin
     match faible q with
       OU_INTRO(x',i0,ET_INTRO (zq))
         -> let zq' j =
              let (z,q_de_j) = zq j in
              let z' = F(X,z) in
              let annot' = (x, z'::X, ajoute (z,z') sigma) in
              (z',COUPE(annot', p, q_de_j)) in
            let p' = ET_INTRO (zq') in
            if x=x' then match p with
                ET_INTRO(yp) ->
                  let (y,p_de_i0) = yp i0 in
                  let annot'' = (y, X, sigma_vide) in
                  faible (COUPE(annot'',p', p_de_i0))
              | _ -> failwith "Preuve mal construite"
            else
                OU_INTRO(associe x' sigma,i0,p')
     | INDETERMINE -> INDETERMINE
     | _ -> failwith "Preuve mal construite"
   end
 | _                         -> p;;
```

Elimination des coupures en suivant la stratégie de réduction de la coupure la plus externe d'abord

```
let rec externe p =
  match (faible p) with
   OU_INTRO (x,i0,p') -> OU_INTRO(x,i0,externe p')
 | ET_INTRO (zp)      -> ET_INTRO(fun j -> (fst (zp j),externe (snd (zp j))))
 | INDETERMINE        -> INDETERMINE
 | _                  -> failwith "Anomalie";;
```

Un exemple concret :

On considère une preuve de l'énoncé suivant : dans toute suite f de 0 et de 1, il existe au moins 2 termes égaux. La preuve est obtenue par coupure entre un lemme classique affirmant que dans la suite, il y a une infinité de 1 ou une infinité de 0, et un lemme intermédiaire affirmant l'existence de 2 termes identiques à partir de l'existence d'une infinité de termes identiques.

Le lemme classique

On pose $A_i(p,q)$ pour $p \leq q \wedge f(q) = i$ avec $i = 0$ ou $i = 1$. La proposition $\forall_{p_i}\exists_{q_i}A_i(p_i,q_i)$ (où $\forall$ et $\exists$ désignent ici des conjonctions et disjonctions infinies) exprime l'existence d'une infinité de termes de f égaux à i.

Plusieurs preuves du lemme classique sont possibles. On en propose une dont la traduction dans le calcul de Novikoff est la suivante (m est le maximum de p_0 et p_1) :

$$
\left[\;\left[\;
\dfrac{
\dfrac{\dfrac{}{;p_1 \leq m}\,Ax}{p_1 \leq m;}\,I_\exists
\qquad
\dfrac{\dfrac{
\dfrac{\dfrac{}{;p_0 \leq m}\,Ax}{p_0 \leq m;}\,I_\exists
\quad
\dfrac{\dfrac{}{;f(m) = f(m)}\,Ax}{f(m) = 1,\ f(m) = 0;}\,I_\exists
}{f(m) = 1;\ p_0 \leq m \wedge f(m) = 0}\,I_\forall}{\exists_{q_0}A_0(p_0,q_0),\ f(m) = 1;}\,I_\exists
}{
\dfrac{\exists_{q_0}A_0(p_0,q_0);\ p_1 \leq m \wedge f(m) = 1}{\exists_{q_1}A_1(p_1,q_1),\ \exists_{q_0}A_0(p_0,q_0);}\,I_\exists
}\,I_\forall
\;\right]_{p_0}
\atop
\dfrac{\dfrac{\exists_{q_1}A_1(p_1,q_1);\ \forall_{p_0}\exists_{q_0}A_0(p_0,q_0)}{\forall_{p_0}\exists_{q_0}A_0(p_0,q_0),\ \exists_{q_1}A_1(p_1,q_1);}\,I_\exists}{}\,I_\forall
\;\right]_{p_1}
$$

$$
\dfrac{\dfrac{\forall_{p_0}\exists_{q_0}A_0(p_0,q_0),\ \exists_{q_1}A_1(p_1,q_1);}{\forall_{p_0}\exists_{q_0}A_0(p_0,q_0);\ \forall_{p_1}\exists_{q_1}A_1(p_1,q_1)}\,I_\forall}{(\forall_{p_0}\exists_{q_0}A_0(p_0,q_0)) \vee (\forall_{p_1}\exists_{q_1}A_1(p_1,q_1));}\,I_\exists
$$

L'idée de la preuve est de "jouer sur deux tableaux" en essayant simultanément de trouver un $q_0 > p_0$ tel que $f(q_0) = 0$ et un $q_1 > p_1$ tel que $f(q_1) = 1$. Pour cela prendre $q_0 = q_1 = max(p_0,p_1)$ est un bon candidat. Pour cette valeur, au moins un des deux côtés de la disjonction $A_0(p_0,m) \vee A_1(p_1,m)$ est vrai.

Remarques : 1) On s'est contenté de ne faire figurer les formules provenant d'un affaiblissement qu'au moment où elles deviennent nécessaires à la preuve.

2) Puisque le langage considéré n'a que des axiomes déterminés, il faut entendre la preuve ci-dessus comme dépendante de f : la dépendance a lieu dans la règle $I_\exists$ précédant l'axiome $\overline{;f(m) = f(m)}$ (c'est le membre vrai qui est choisi).

La preuve, traduite dans les structures ML précédentes, est celle-ci :

```
let prefix ! = OU_INTRO
and prefix ? = ET_INTRO;;

let Axiome = ? (fun O -> ("bidon",failwith "Pas de prémisse à un axiome"));;
```

```
let max (q0,q1) =
  if q1 > q0 then q1 else q0;;

let lemme_classique f =
  ! ("inf0_ou_inf1", B true,
   ? (fun (E p0) -> ("inf0(p0)",
    ! ("inf0_ou_inf1", B false,
     ? (fun (E p1) -> ("inf1(p1)",
      let m = max(p0,p1) in
      ! ("inf0(p0)", E m,
       ? (fun
         (B true)  -> ("m>p0",
           ! ("m>p0",S,Axiome))
       | (B false) -> ("f(m)=0",
           ! ("inf1(p1)", E m,
            ? (fun
              (B true)  -> ("m>p1",
                ! ("m>p1",S,Axiome))
            | (B false) -> ("f(m)=1",
                if f(m)=0
                  then ! ("f(m)=0", E m, Axiome)
                  else ! ("f(m)=1", E m, Axiome))))))))))));;
```

Le lemme intermédiaire

Posons $B = \exists_{a,b,i}(a<b) \land (f(a)=i) \land (f(b)=i)$.
La proposition B exprime l'existence de deux termes identiques dans f.
La négation du prédicat A s'écrit $A_i^\perp(p,q) = (q < p \lor f(q) \neq i)$.
Ci-dessous, "*" est soit "<", soit "≥", selon la valeur de vérité de $q_i < q_i'$.

$$\left[\dfrac{\dfrac{\left[\dfrac{\dfrac{\dfrac{\dfrac{}{;\, q_i *^\perp q_i'}\,Ax}{q_i'<q_i+1 \lor f(q_i')\neq i,\ q_i<q_i';}\,I_\exists \quad \dfrac{\dfrac{}{;\, f(q_i)=f(q_i)}\,Ax}{q_i<0 \lor f(q_i)\neq i,\ f(q_i)=i;}\,I_\exists \quad \dfrac{\dfrac{}{;\, f(q_i')=f(q_i')}\,Ax}{q_i'<q_i+1 \lor f(q_i')\neq i,\ f(q_i')=i;}\,I_\exists}{(q_i<0 \lor f(q_i)\neq i),\ (q_i'<q_i+1 \lor f(q_i')\neq i);\ (q_i<q_i') \land (f(q_i)=i) \land (f(q_i')=i)}\,I_\forall}{B,\ (q_i<0 \lor f(q_i)\neq i),\ (q_i'<q_i+1 \lor f(q_i')\neq i);}\,I_\exists\right]_{q_i'}}{\dfrac{B,\ (q_i<0 \lor f(q_i)\neq i);\ \forall_{q_i'} A_i^\perp(q_i+1,q_i')}{\exists_{p_i'}\forall_{q_i'} A_i^\perp(p_i',q_i'),\ B,\ (q_i<0 \lor f(q_i)\neq i);}\,I_\exists}\,I_\forall\right]_{q_i}$$

$$\dfrac{\exists_{p_i'}\forall_{q_i'} A_i^\perp(p_i',q_i'),\ B;\ \forall_{q_i} A^\perp(0,q_i)}{\exists_{p_i}\forall_{q_i} A_i^\perp(p_i,q_i),\ B;}\,I_\exists$$

La formule $\exists_{p_i}\forall_{q_i} A_i^\perp(p_i,q_i)$ exprime l'hypothèse d'une infinité de termes valant i (c'est-à-dire qu'après tout p_i, il y a un q_i tel que $f(q_i)=i$).

L'idée de la preuve est alors de donner la valeur 0 à p_i dans l'attente d'un tel $q_i \geq 0$ vérifiant $f(q_i)=i$, puis de recommencer le même procédé en donnant la valeur q_i+1 à p_i dans l'attente d'un nouveau $q_i' \geq q_i+1$ tel que $f(q_i')=i$.

En exhibant le triplet (q_i, q_i', i), on est sur le point de prouver le lemme. Ou bien les q_i et q_i' obtenus satisfont ce que l'on attend d'eux auquel cas les conditions de la vérité de B sont réunies, ou alors, on réfute l'hypothèse que q_i et q_i' étaient "corrects".

A cette preuve correspond le codage suivant en ML :

```
let lemme_intermédiaire f i =
  ! ("~inf", E 0,
   ? (fun (E q) -> ("~A(0,q)",
    ! ("~inf", E (q+1),
     ? (fun (E q') -> ("~A(p+1,q')",
      ! ("bin", T (E q,E q',E i),
       ? (fun
       (D Gauche) -> ("q'>q",
         if (q'>q)
          then ! ("q'>q"      , S, Axiome)
          else ! ("~A(p+1,q')", B true, Axiome))
     | (D Milieu) -> ("f(q)=i",
         if f(q)=i
          then ! ("f(q)=i" , S , Axiome)
          else ! ("~A(0,q)", B false, Axiome))
     | (D Droite) -> ("f(q')=i",
         if f(q')=i
          then ! ("f(q')=i"   , S , Axiome)
          else ! ("~A(p+1,q')", B false, Axiome))))))))));;
```

La coupure entre les lemmes

$$\dfrac{\dfrac{\begin{array}{cc}\text{lemme intermédiaire} & \text{lemme intermédiaire}\\ \text{pour } 0 & \text{pour } 1\\ \vdots & \vdots \\ B, \exists_{p_0}\forall_{q_0} A_0^{\perp}(p_0,q_0); & B, \exists_{p_1}\forall_{q_1} A_1^{\perp}(p_1,q_1);\end{array}}{B; (\exists_{p_0}\forall_{q_0} A_0^{\perp}(p_0,q_0)) \wedge (\exists_{p_1}\forall_{q_1} A_1^{\perp}(p_1,q_1))}\, I_{\forall} \qquad \begin{array}{c}\text{lemme classique}\\ \vdots \\ (\forall_{p_0}\exists_{q_0} A_0(p_0,q_0)) \vee (\forall_{p_1}\exists_{q_1} A_1(p_1,q_1));\end{array}}{B}\, \mathit{Coupe}$$

La représentation de cette coupure dans notre application ML est :

```
let coupure f =
  COUPE (("inf0_ou_inf1",["inf0_ou_inf1";"bin"],[]),
    ? (fun
      (B true)  -> ("~inf",lemme_intermédiaire f 0)
    | (B false) -> ("~inf",lemme_intermédiaire f 1)),
    lemme_classique f);;
```

L'élimination de la coupure

Sur deux exemples de suites, on obtient les résultats suivants :

```
let f1 n = n mod 2;;

let f2 n = (n + 1) mod 2;;

#externe (coupure f1);;
- : preuve = OU_INTRO ("bin", D (E 0, E 2, E 0), ET_INTRO <fun>)

#externe (coupure f2);;
- : preuve = OU_INTRO ("bin", D (E 1, E 3, E 0), ET_INTRO <fun>)
```

Et si l'on analyse ce qui se cache derrière le `<fun>`, on trouve, dans les deux cas :

```
fun D Gauche -> ("q'>q###"   , OU_INTRO ("q'>q###"   , S, Axiome))
 |  D Milieu -> ("f(q)=i###" , OU_INTRO ("f(q)=i###" , S, Axiome))
 |  D Droite -> ("f(q')=i###", OU_INTRO ("f(q')=i###", S, Axiome))
```

Remarque : On retrouve ici un aspect clef de cette preuve : `f1` et `f2` sont symétriques l'une de l'autre par échange entre les "0" et les "1", mais les résultats de l'exécution de la procédure `externe` de le sont pas (cf à ce sujet [9, 10]).

Débat entre preuves vues comme stratégies gagnantes

La représentation des stratégies

Comme précédemment, c'est le type `index` qui servira à indexer les conjonctions et les disjonctions.

Les occurrences de sous-formules dans une formule sont des listes d'indices et les positions sont des listes d'occurrences. Non exprimées par le typage, les propriétés qu'une position doit vérifier sont supposées controlées de manière externe.

```
type occurrence == index list;;

type position == occurrence list;;
```

Par commodité, puisque $\phi(u_1...u_n)$ est l'extension directe $u_k i$ d'un seul des u_k, on exprime $\phi(u_1...u_n)$ par le couple (k, i).

De plus, plutôt que de se restreindre à des preuves partielles s'affrontant sur des formules duales l'une de l'autre, on considère aussi des preuves qui, au bout d'un certain temps, quittent le débat autour de la formule de coupure pour jouer un coup dans une formule autre que la formule de coupure. C'est le rôle du constructeur `QUITTE`.

```
type coup =
   REPOND of int * index
 | QUITTE of nom * index
 | INDEFINI;;

type stratégie == position -> coup;;
```

Une partie est une suite de coups justifiés, c'est-à-dire une suite constituée de paires (f, u) où f est une référence à un coup précédent et u une référence à une sous-formule de la formule disputée. On laisse implicite la contrainte qu'une partie est non vide.

```
type partie == (int * occurrence) list;;
```

Pour $k < n$, cette fonction retourne les couples $(f_0, u_0)...(f_k, u_k)$ d'une partie $(f_0, u_0)...(f_n, u_n)$.

```
let restreint partie k =
  let rec début segment_initial k partie =
    if k=0 then segment_initial
    else match partie with
        []       -> []
      | a::reste -> début (a::segment_initial) (k-1) reste
  in
    match (rev partie) with
      c::reste -> début [c] k reste
    | [] -> failwith "Anomalie";;
```

Ici, on extrait d'une partie la position courante. On garde mémoire de l'indice de chaque coup dans la partie.

```
let rec position_indexée partie =
  let n = (list_length partie) - 1 in
  match partie with
   (f,occ)::reste
      -> if f=0 then [(n,occ)]
                else (position_indexée (restreint reste (f-1)))@[(n,occ)]
 | [] -> failwith "Anomalie";;
```

La fonction `position` oublie les indices associés aux occurrences de la position courante.

```
let position pos_ind = map snd pos_ind;;
```

Ici, on calcule le nouveau coup joué en fonction des coups déjà joués (`partie`) et du joueur dont c'est le tour de jouer (`joueur`). Si le joueur s'arrête de jouer, une exception `Terminé` est levée.

```
exception Terminé of partie * coup;;

let rec n_ième k = function
   a::l -> if k=1 then a else (n_ième (k-1) l)
 | []   -> failwith "Référence incorrecte";;

let ajoute_coup (joueur:stratégie) (partie:partie) =
  let pi = position_indexée partie in
  let réponse = joueur (position pi) in
  match réponse with
    REPOND (k,i) -> let (f,ancien_coup) = n_ième k pi in
                    let nouveau_couple = (f,i::ancien_coup) in
                    nouveau_couple::partie
  | _           -> raise (Terminé (rev partie,réponse));;
```

Le débat fait alterner les coups de `phi` et `psi`.

```
let débat (phi:stratégie) (psi:stratégie) =
  let rec phi_joue partie = psi_joue (ajoute_coup phi partie)
      and psi_joue partie = phi_joue (ajoute_coup psi partie)
  in
  let partie_initiale = [(0,[])] in
    try phi_joue partie_initiale
    with Terminé (partie,réponse) -> (partie,réponse);;
```

La traduction d'une preuve en une stratégie

La traduction passe par l'association d'un nœud de la preuve à toute position bien définie.

```
let ordre x = ordre_rec 1
  where rec ordre_rec n = function
   [] -> failwith "Nom incorrect"
 | y::l -> if x=y then n else ordre_rec (n+1) l;;

let rec noeud l = function
   (occ::pos,OU_INTRO (x,i,ET_INTRO (zq))) -> let j = hd occ in
                                              let (z,q_de_j) = zq j in
                                              noeud (l@[z]) (pos,q_de_j)
 | ([],OU_INTRO (x,i,_)) -> (try REPOND (ordre x l,i) with _ -> QUITTE (i,x))
 | (_,INDETERMINE)       -> INDEFINI
 | (_,COUPE _)           -> failwith "Preuve non normale"
 | _                     -> failwith "Preuve mal construite";;
```

```
let stratégie p =
  fun   []      -> failwith "Ce n'est pas une position correcte"
   | (occ::pos) -> match p with
       OU_INTRO (x,_,_) -> noeud [x] (pos,p)
     | ET_INTRO (zp)    -> let i = hd occ in
                           let (z,p_de_i) = zp i in
                           noeud [z] (pos,p_de_i);;
```

Exemple

On reprend l'exemple précédent de l'existence de deux termes identiques dans une suite de 0 et de 1. Le débat se déroule entre une stratégie gagnante pour le lemme classique $\forall_{p_0}\exists_{q_0}A_0(p_0,q_0)) \vee (\forall_{p_1}\exists_{q_1}A_1(p_1,q_1))$ et une stratégie gagnante pour la paire des lemmes intermédiaires $(\exists_{p_0}\forall_{q_0}A_0^{\perp}(p_0,q_0) \wedge \exists_{p_0}\forall_{q_0}A_i^{\perp}(p_0,q_0)),\ B$. On considère des versions simplifiées des preuves précédentes de telle sorte qu'un coup dans B signifie la fin de la partie. Pour cela on prend B, comme une simple disjonction de formules vraies :

```
let lemme_classique f =
  ! ("inf0_ou_inf1", B true,
   ET_INTRO (fun (E q0) -> ("inf0(q0)",
    ! ("inf0_ou_inf1", B false,
     ? (fun (E q1) -> ("inf1(q1)",
      let m = max(q0,q1) in
      if f(m)=0 then ! ("inf0(q0)", E m, Axiome)
                else ! ("inf1(q1)", E m, Axiome)))))));;

let lemme_intermédiaire f i =
  ! ("~inf", E 0,
   ? (fun (E q) -> ("~inf(0)(q)",
    ! ("~inf", E (q+1),
     ? (fun (E q') -> ("~inf(q+1)(q')",
      ! ("bin", T (E q,E q',E i), Axiome)))))));;

let produit_lemmes_intermédiaires f =
  ? (fun
     (B true)  -> ("~inf",lemme_intermédiaire f 0)
   | (B false) -> ("~inf",lemme_intermédiaire f 1));;

let phi f = stratégie (lemme_classique f);;

let psi f = stratégie (produit_lemmes_intermédiaires f);;
```

A titre indicatif, on donne quelques valeurs de `phi f1` et `psi f1` :

```
#phi f1 [[]];;
- : coup = REPOND (1, B true)
#phi f1 [[];[E 3;B true]];;
- : coup = REPOND (1, B false)
#phi f1 [[];[E 3;B true];[E 4;B false]];;
- : coup = REPOND (2, E 4))
#phi f1 [[];[E 3;B true];[E 5;B false]];;
- : coup = REPOND (3, E 5))

#psi f1 [[B true]];;
```

```
- : coup = REPOND (1, E 0)
#psi f1 [[B true];[E 4;B true]];;
- : coup = REPOND (1, E 5)
#psi f1 [[B true];[E 4;B true];[E 7;B true]];;
- : coup = QUITTE (T (E 4, E 7, E 0))
```

Le débat est bien correct :

```
#débat (phi f1) (psi f1);;
- : partie * coup =
[0, []; 0, [B true]; 1, [E 0; B true]; 0, [B false]; 3, [E 0; B false];
 2, [E 0; E 0; B true]; 1, [E 1; B true]; 0, [B false]; 7, [E 0; B false];
 8, [E 1; E 0; B false]; 7, [E 2; B false]; 6, [E 2; E 1; B true]],
QUITTE ("bin", T (E 0, E 2, E 0))

#débat (phi f2) (psi f2);;
- : partie * coup =
[0, []; 0, [B true]; 1, [E 0; B true]; 0, [B false]; 3, [E 0; B false];
 4, [E 0; E 0; B false]; 3, [E 1; B false]; 2, [E 1; E 0; B true];
 1, [E 2; B true]; 0, [B false]; 9, [E 0; B false]; 10, [E 2; E 0; B false];
 9, [E 3; B false]; 8, [E 3; E 2; B true]],
QUITTE ("bin", T (E 1, E 3, E 0))
```

Remarque : Au niveau des calculs, il y a moyen de faire des simplifications. Notamment, il est inutile de garder l'information des occurrences. Seule l'information du dernier index constituant de l'occurrence suffit au bon déroulement du calcul.

Normalisation en $\overline{\lambda}$-calcul

Une procédure de normalisation

On transcrit en ML la procédure de mise en forme normale du $\overline{\lambda}$-calcul (ici, c'est la stratégie de réduction propre du ML considéré qui détermine la stratégie de normalisation des $\overline{\lambda}$-expressions ; en l'occurrence, avec Caml-Light, langage qui est adopté ici, la normalisation se fera par réduction des rédex les plus internes d'abord).

On choisit comme variables des chaînes de caractères et on se donne le moyen d'engendrer de nouvelles variables en les "primant".

```
type variable == string;;

let prime y = y^"'";;
```

La syntaxe des $\overline{\lambda}$-termes met en jeu termes et listes de termes.

```
type lb_terme =
   Appx of variable * lb_liste_terme
 | Abs of variable * lb_terme
 | Appt of lb_terme * lb_liste_terme
 | Subt of lb_terme * variable * lb_terme
and lb_liste_terme =
   Vide
 | Cons of lb_terme * lb_liste_terme
 | Conc of lb_liste_terme * lb_liste_terme
 | Subl of lb_liste_terme * variable * lb_terme;;
```

On se donne des procédures de renommage des variables liées.

```
let rec nouveau y l =
  if mem y l then nouveau (prime  y) l
             else y;;

let rec renomme_terme l y y' = function
     Appx (x,lt)  ->
      if x=y then Appx (y',renomme_liste_terme l y y' lt)
             else Appx (x,renomme_liste_terme l y y' lt)
  |  Abs (x,t)   ->
      let x' = nouveau x (y::l) in
      let t' = if x=x' then t else renomme_terme (y::l) x x' t in
      Abs (x', renomme_terme (x'::l) y y' t')
  | _ -> failwith"Anomalie : terme non normal"

and renomme_liste_terme l y y' = function
    Cons (t,lt) -> Cons (renomme_terme l y y' t, renomme_liste_terme l y y' lt)
  | Vide        -> Vide
  | _ -> failwith"Anomalie : liste de termes non normale";;
```

On donne les procédures de réduction des différents types de rédex. Chaque constructeur de rédex prend en compte un argument supplémentaire consistant en une liste de variables censée englober toutes les variables libres des arguments considérés (typiquement, dans le cadre simplement typé, cette liste est la liste des noms dans le "Γ" du jugement $\Gamma; \Pi \vdash A$). Ceci constitue une solution "propre" au problème d'α-conversion et de confluence en présence d'α-conversion.

```
let rec appt l (t,lt) =
  match t with
    Appx (x,lt') -> Appx (x,conc l (lt', lt))
  | Abs (x,t')    ->
      begin
        match lt with
           Vide       -> t
         | Cons (u,lt') -> appt l (subt (x::l) (t', x, u), lt')
         | _          -> failwith"Anomalie : terme non normal"
      end
  | _            -> failwith"Anomalie : terme non normal"

and subt l (t,x,u) =
  match t with
    Appx (y,lt) ->
      if x=y then
        appt l (u, subl l (lt,x,u))
      else
        Appx (y, subl l (lt,x,u))
  | Abs (y,t')   ->
      let y' = nouveau y l in
      let t'' = if y=y' then t' else renomme_terme l y y' t' in
      Abs (y', subt (y'::l) (t'',x,u))
  | _           -> failwith"Anomalie : terme non normal"

and conc l (lt,lt') =
  match lt with
    Cons (t,lt'') -> Cons (t, conc l (lt'', lt'))
  | Vide          -> lt'
  | _             -> failwith"Anomalie : liste de termes non normale"

and subl l (lt,x,u) =
  match lt with
    Cons (t,lt') -> Cons (subt l (t, x, u), subl l (lt',x,u))
  | Vide         -> Vide
  | _            -> failwith"Anomalie : liste de termes non normale";;

let rec terme_normal l = function
  Appx (x,lt)    -> Appx (x, liste_terme_normal l lt)
| Abs (x,t)      -> Abs (x, terme_normal (x::l) t)
| Appt (t,lt)    -> appt l (terme_normal l t, liste_terme_normal l lt)
| Subt (t,x,u) -> subt l (terme_normal (x::l) t, x, terme_normal l u)

and liste_terme_normal l = function
  Vide            -> Vide
| Cons (t,lt)     -> Cons (terme_normal l t, liste_terme_normal l lt)
| Conc (lt,lt') -> conc l (liste_terme_normal l lt,
                           liste_terme_normal l lt')
| Subl (lt,x,t) -> subl l (liste_terme_normal (x::l) lt,
                           x,
                           terme_normal l t);;
```

Exemple

D'abord un traducteur élémentaire de syntaxe interne en syntaxe concrète. On se restreint à l'affichage de termes sans opérateurs ni de substitution ni de concaténation.

```
let rec affiche_terme = function
   Appx (x,Vide)  -> x
 | Abs (x,t)      -> "\\"^x^(affiche_reste_abs t)
 | t              -> (affiche_app_terme t)

and affiche_reste_abs = function
   Abs (x,t)      -> ","^x^(affiche_reste_abs t)
 | Appx (x,Vide)  -> "."^x
 | t              -> ".("^(affiche_app_terme t)^")"

and affiche_app_terme =  function
   Appx (x,lt)    -> x^(affiche_liste_terme lt)
 | Appt (t,lt)    -> (affiche_app_terme t)^(affiche_liste_terme lt)
 | Abs (x,t)      -> "(\\"^x^(affiche_reste_abs t)^")"

and affiche_liste_terme = function
   Vide           -> "[]"
 | Cons (t,lt)    -> "["^(affiche_terme t)^(affiche_reste_liste_terme lt)^"]"

and affiche_reste_liste_terme = function
   Vide           -> ""
 | Cons (t,lt)    -> ";"^(affiche_terme t)^(affiche_reste_liste_terme lt)
;;
```

et un traducteur inverse très sommaire (incluant la gestion d'un environnement de constantes) :

```
let env = ref ([] : (string * lb_terme) list);;

let rec reste_ident = function
  [< '(`a`..`z` | `0`..`9` | `'` | `é` (* ... *) as c); reste_ident s >] -> (make_string 1 c)^s
  | [< >] -> "";;

let lit_variable = function
  [< '(`a`..`z` as c); reste_ident s >] -> (make_string 1 c)^s

let lit_constante = function
  [< '(`A`..`Z` | `0`..`9` as c); reste_ident s >] -> (make_string 1 c)^s

let rec saute_blancs = function
  [< '(` ` | `\n`); saute_blancs s >] -> s
| [< 'c; saute_blancs s >] -> [< 'c; s >]
| [< >] -> [< >];;

let rec applique = fun
  (Appx (x,Vide),[l])       -> Appx (x,l)
| (Appx (x,Vide),l::l'::lr) -> applique (Appt(Appx(x,l),l'),lr)
| (t,l::lr)                 -> applique (Appt(t,l),lr)
| (t,[])                    -> t;;

let rec lit_terme = function
  [< lit_abs_terme t >]                           -> t
| [< lit_terme_simple t; lit_arguments llt >]     -> applique (t,llt)

and lit_terme_simple = function
```

```
  [< '`(`; lit_terme t; '`)` >]                       -> t
| [< lit_variable x >]                                -> Appx (x,Vide)
| [< lit_constante c >]                               -> assoc c !env

and lit_arguments = function
  [< '`[`; lit_termes lt; '`]`; lit_arguments llt >] -> lt::llt
| [< >]                                               -> []

and lit_abs_terme = function
  [< '`\\`; lit_variable x; lit_abs_vars t >]        -> Abs (x,t)

and lit_abs_vars = function
  [< '`.`; lit_reste_abs_terme t >]                   -> t
| [< '`,`; lit_variable x; lit_abs_vars t >]         -> Abs (x,t)

and lit_reste_abs_terme = function
  [< lit_abs_terme t >]                               -> t
| [< lit_terme_simple t >]                            -> t

and lit_termes = function
  [< lit_terme t; lit_reste_termes lt >]              -> Cons (t,lt)
| [< >]                                               -> Vide

and lit_reste_termes = function
  [< '`;`; lit_terme t; lit_reste_termes lt >] -> Cons (t,lt)
| [< >]                                        -> Vide
;;

let terme s = lit_terme (saute_blancs (stream_of_string s));;
```

```
let définit nom s = env := (nom,terme s)::!env;;
```

Enfin, des exemples de termes :

```
définit "K" "\x,y.x";;
définit "K'" "\x,y.y";;
définit "Z" "\f,x.x";;
définit "Delta" "\x.(x[x])";;
définit "Indéfini" "Delta[Delta]";;
définit "Succ" "\m.\f,x.(f[(m[f;x])])";;
définit "Paire" "\a,b.\f.(f[a;b])";;
définit "Proj1" "\p.(p[K])";;
définit "Proj2" "\p.(p[K'])";;
définit "Pompe" "\p.(Paire [(Succ[Proj1[p]]);(Proj1[p])])";;
définit "Fond" "Paire[Z;Z]";;
définit "Pred" "\m.(Proj2[(m[Pompe;Fond])])";;
définit "Egal0" "\m.(m[\x.K;K'])";;
définit "If" "\c.\v1,v2.(c[v1;v2])";;
définit "Pred'" "\m.(If[Egal0[m];Indéfini;Pred[m]])";;
définit "4" "Succ[Succ[Succ[Succ[Z]]]]";;
```

Et un exemple de réduction :

```
#print_string (affiche_terme (terme_normal [] (terme "(Pred [4])")));;
\f'',x'.(f''[f''[f''[x']]])
- : unit = ()
```

Traductions entre le λ-calcul et le $\overline{\lambda}$-calcul

```
type l_terme =
   Var of variable
 | Abs' of variable * l_terme
 | App of l_terme * l_terme;;
```

Traduction littérale

Cette correspondance plonge le λ-calcul dans le $\overline{\lambda}$-calcul. La structure des λ-termes est préservée par la traduction.

```
let rec traduction_littérale = function
   Var x      -> Appx (x, Vide)
 | Abs' (x,t) -> Abs (x, traduction_littérale t)
 | App (t,u)  -> Appt (traduction_littérale t,
                       Cons (traduction_littérale u, Vide));;
```

Traduction morale

Cette traduction met en correspondance les formes normales du λ-calcul et les formes normales du $\overline{\lambda}$-calcul.

```
let rec traduction_morale = function
      Abs' (x,t) -> Abs (x, traduction_morale t)
    | t          -> tradrec t Vide
 where rec tradrec l = function
   Var x         -> Appx (x, l)
 | Abs' (x,t)    -> Appt (Abs (x, traduction_morale t), l)
 | App (t,u)     -> tradrec t (Cons (traduction_morale u,l));;
```

Traduction inverse

Cette traduction est la réciproque de la traduction des formes normales du λ-calcul en formes normales du $\overline{\lambda}$-calcul.

```
let rec traduction_inverse = function
   Abs  (x,t) -> Abs (x, traduction_inverse t)
 | Appx (x,l) -> applique (Var x) l
 | _ -> failwith "Traduction inverse non possible"
and applique t = function
   Vide -> t
 | Cons (u,l) -> applique (LApp (t,u)) l;;
 | _ -> failwith "Traduction inverse non possible";;
```

Bibliographie

[1] S. Abramsky, R. Jagadeesan, P. Malacaria, *Full abstraction for PCF (extended abstract)*, Proceedings of TACS'94, LNCS 789, Springer, pp 1-15, disponible aussi par ftp anonyme à theory.doc.ic.ac.uk

[2] Z. Benaissa, D. Briaud, P. Lescanne, J. Rouyer-Degli, *$\lambda\upsilon$: a calculus of explicit substitutions which preserves strong normalisation*, submitted to the Journal of Functional Programming, 1994.

[3] F. Barbanera, S. Berardi, *A symmetric lambda calculus for "classical" program extraction*, Proceedings of TACS'94, LNCS 789, Springer, pp 495-515.

[4] A. R. Blass, *Degrees of indeterminacy of games*, Fundamenta Mathematicae, Vol 77, 1972, pp 151-166.

[5] A. R. Blass, *A game semantics for linear logic*, Annals of Pure and Applied Logic, Vol 56, 1992, pp 183-220.

[6] N. G. de Bruijn, *A survey of the project Automath*, dans "To H.B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism", édité par J.P. Seldin et J.R. Hindley, Academic Press, 1980.

[7] V. Breazu Tanen, D. Kesner, L. Puel, *A typed pattern calculus*, Proceedings, Eighth Annual IEEE Symposium on Logic in Computer Science (LICS'93), IEEE Computer Society Press, pp 262-274.

[8] A. Church, *A set of postulates for the foundation of logic*, Annals of Mathematics, Vol 33, 1932, pp 346-366, et Vol 34, 1933, pp 839-864.

[9] T. Coquand, *A semantics of evidence for classical arithmetic, revised version*, à paraître dans Journal of Symbolic Logic. Première version dans : Proceedings of the CLICS workshop, Århus, 1992.

[10] T. Coquand, H. Herbelin, *Some remarks on Novikoff's calculus*, article privé, 1994.

[11] P.-L. Curien, *An abstract framework for environment machines*, Theoretical Computer Science, Vol 82, 1991, pp 389-402.

[12] H. B. Curry, *Grundlagen der kombinatorische Logik*, American Journal of Mathematics, Vol 52, 1930, pp 509-536 et pp 789-834.

[13] H. B. Curry, R. Feys, W. Craig, *Combinatory Logic, Tome 1*, North-Holland, 1958, §9E.

[14] V. Danos, J.-B. Joinet, H. Schellinx, *LKQ and LKT : sequent calculi for second order logic based upon dual linear decompositions of classical implication*, à paraître dans les actes du "Workshop on Linear Logic", Cornell, edited by J.-Y. Girard, Y. Lafont, L. Régnier, 1993.

[15] A. G. Dragalin, *Mathematical Intuitionism : Introduction to Proof Theory*, Translations of mathematical monographs 67, Providence, R.I. : American Mathematical Society, 1988, l'édition originale en russe datant de 1979.

[16] M. Felleisen, *Reasoning about continuations*, Proceedings, First Annual IEEE Symposium on Logic in Computer Science (LICS'86), IEEE Computer Society Press, pp 131-141.

[17] W. Felscher, *Dialogues as a foundation of intuitionistic logic*, Handbook of Philosophical Logic, Vol 3, pp 341-372, 1986.

[18] W. Felscher, *Dialogues, strategies, and intuitionistic provability* Annals of Pure and Applied Logic, Vol 28, pp 217-254, 1985.

[19] , J. Gallier, *Constructive logics, part I: a tutorial on proof systems and typed λ-calculi*, Theoretical Computer Science, Vol 110, 1993, pp 249-339.

[20] G. Gentzen, *Investigations into logical deduction*, cf par exemple dans [55], pp 68ff.

[21] G. Gentzen, *The consistency of elementary number theory* cf par exemple dans [55], pp 132ff.

[22] G. Gentzen, *New version of the consistency proof for elementary number theory*, cf par exemple dans [55], pp 252ff.

[23] J.-Y. Girard, *Linear logic*, Theoretical Computer Science, Vol 50, 1987, pp 1-102.

[24] J.-Y. Girard, *Proof Theory and Logical Complexity, Tome 1*, Bibliopolis, 1987.

[25] J.-Y. Girard, *A New constructive logic : classical logic*, Mathemathical Structures in Computer Science, Vol 1, 1991, pp 255-296.

[26] J.-Y. Girard, *On the unity of logic*, Annals of Pure and Applied Logic, Vol 59, 1993, pp 201-217.

[27] J.-Y. Girard, Y. Lafont, P. Taylor, *Proofs and Types*, Cambridge University Press, 1989.

[28] G. Gonthier, J.-J. Lévy, P.-A. Méllies, *A standardisation theorem*, Proceedings, Seventh Annual IEEE Symposium on Logic in Computer Science (LICS'92), IEEE Computer Society Press, pp 72-81.

[29] T. Griffin, *A formulae-as-types notion of control*, Conference record of the Seventeenth Annual ACM Symposium on Principles of Programing Languages (Actes de POPL'90), Association for Computing Machinery, pp 47-58.

[30] T. Hardin, J.-J. Lévy, *A confluent calculus of substitutions*, dans "France-Japon artificial intelligence and computer science symposium", 1989.

[31] W. A. Howard, *The formulae-as-types notion of constructions*, dans "to H.B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism", édité par J.P. Seldin et J.R. Hindley, Academic Press, 1980 (Manuscrit non publié de 1969).

[32] G. Huet, *Confluent reductions : abstract properties and applications to term rewriting systems*, Journal of the Association for Computing Machinery, Vol 27, 1980, pp 797-821.

[33] M. Hyland, L. Ong, *Dialogue games and innocent strategies : an approach to (intensional) full abstraction for PCF, preliminary announcement*, disponible par ftp anonyme à theory.doc.ic.ac.uk.

[34] I. Johansson, *Der Minimalkalkül, ein reduzierter intuitionistischer Formalismus*, Compositio Mathematica, Vol 4, 1937, pp 119-136.

[35] J.-B. Joinet, *Etude de la normalisation du calcul des séquents classique à travers la logique linéaire*, thèse de doctorat, université Paris VII, 1993.

[36] K. Lorenzen, *Algebraische und logistische Untersuchung über freie Verbände*, Journal of Symbolic Logic, Vol 16, N 2, 1951, pp 81-106, critique en anglais dans Vol 16, N 4, 1951, pp 269-272.

[37] P. Lorenzen, *Einführung in die operative Logik und Mathematik*, Springer, 1955, p 5.

[38] P. Lorenzen, *Logik und Agon*, in Atti Congr. Internat. di Filosofia, Vol 4, Sansoni, Firenze, 1960, pp 187-194.

[39] P. Lorenzen, *Ein dialogisches Konstruktivitätkriterium*, Proceedings of the Symposium on Foundations of Mathematics, PWN, Warszawa, 1961, pp 193-200.

[40] P. Lorenzen, *Métamathématique*, Collection Mathématiques et sciences de l'homme, Vol 6, édité par Mouton, 1967 (édition originale en allemand de 1962).

[41] P. Martin-Löf, *Notes on constructive mathematics*, Almqvist & Wiksell, Stockholm, 1968, §30.

[42] P. Martin-Löf, *Intuitionistic type theory*, Bibliopolis, 1988.

[43] G. Mints, *Normal forms for sequent derivations*, communication privée.

[44] Y. N. Moschovakis, *A model of concurrency with fair merge and full recursion*, Information and Computation, Vol 93, 1991, pp 114-171.

[45] C. Murthy, *Extracting constructive content from classical proofs*, Ph. D. Thesis, Cornell University, 1990.

[46] P. S. Novikoff, *On the consistency of certain logical calculus*, Matematiceskij sbornik (Recueil Mathématique), Vol 12 (54), N 2, 1941, pp 230-260.

[47] M. J. O'Donnell, *Computing in Systems Described by Equations*, LNCS 58, Springer, 1977.

[48] M. Parigot, *$\lambda\mu$-calculus: an algorithmic interpretation of classical Natural Deduction*, LNCS 624, Springer, pp 190-201.

[49] G. Pottinger, *Normalization as a homomorphic image of cut-elimination*, Annals of mathematical logic (rebaptisé Annals of Pure and Applied Logic), Vol 12, 1977, pp 323-357.

[50] D. Prawitz, *Natural deduction, a proof-theoretical study*, Almquist & Wiksell, 1965, Stockholm.

[51] H. Schellinx, *The Noble Art of Linear Decorating*, ILLC Dissertation Series 1994-1, Institute for Language, Logic and Computation, University of Amsterdam, 1994.

[52] M. Schönfinkel, *Über die Bausteine der mathematische Logik*, Mathematische Annalen, Vol 92, 1925, pp 305-316.

[53] K. Schütte, *Schlußweisen-Kalküle der Prädikatenlogic*, Mathematische Annalen, Vol 122, 1950, pp 47-65.

[54] H. Schwichtenberg, *Proof Theory: Some Applications of Cut-Elimination*, Handbook of Mathematical Logic, édité par J. Barwise, North Holland, 1977, pp 867-895.

[55] M. E. Szabo, *Gentzen collected work*, North Holland, 1969.

[56] E. Tahhan Bittar, *Gentzen cut elimination for propositional calculus by rewriting derivations*, publication du Laboratoire de Logique, d'Algorithmique et d'Informatique de Clermont I, numéro 16, 1992.

[57] W. W. Tait, *Normal derivability in classical logic*, Lecture Notes in Mathematics 72, édité par J. Barwise, 1968, pp 204-236.

[58] A. S. Troelstra, *Lectures in Linear Logic*, CSLI Lecture Notes, numéro 29, 1991.

[59] P. Wadler, *A Curry-Howard isomorphism for sequent calculus*, communication privée, 1993.

[60] J. I. Zucker, *Correspondence between cut-elimination and normalization, part I and II*, Annals of mathematical logic (rebaptisé Annals of Pure and Applied Logic), Vol 7, 1974, pp 1-156.

Table des matières